Universidad
Carlos III de Madrid

TRABAJO FIN DE GRADO

Título: Microtestbed integrado para desarrollo y distribución de medidas en Internet
Autor: Adrián Pantoja Navarro
Titulación: Grado en Ingeniería Telemática
Tutor: Francisco Valera Pintor
Fecha: 21 de Junio de 2013

Agradecimientos

Antes de nada, quisiera dar las gracias a mi tutor, Francisco Valera, sin el cual este proyecto no hubiera sido lo que ha llegado a ser.

Posiblemente tengan ustedes delante el resultado del mayor logro personal en toda mi vida. Tras largos años de esfuerzo y sacrificio, el fruto ha surgido y una etapa ha llegado a su fin. A lo largo de estos siete años han pasado muchísimas cosas y hoy me siento agradecido por las grandes personas que me rodean, y que me han apoyado en los momentos más difíciles a lo largo de todo el recorrido.

Podría citar nombres pero sería injusto por todos aquellos que se quedaran fuera de esta hoja, la memoria falla cuando más te puede hacer falta. Agradecido a todas las personas que me han apoyado, que han creído en mí, a mis amigos, a la mujer de mi vida y sobre todo a mi familia.

También me querría acordar de aquellos que se han quedado en el camino. Gracias a mi abuela Encarna por enseñarme el valor de luchar, recordar tu sonrisa siempre me da fuerza para pelear por lo que deseo. Gracias abuelo Félix por enseñarme las cosas que realmente tienen valor en esta vida. Gracias tía-abuela, como siempre te gustaba que te llamara, por mostrarme el camino de la perseverancia y de la lucha contra el día a día. Aunque hoy ya no estéis conmigo siempre estaréis en mi corazón.

Para terminar, agradecer especialmente todo este trabajo a dos personas: mi madre y mi hermano. Sin vosotros, con vuestro cariño y vuestro apoyo, esto hubiera sido inalcanzable. Sois, y siempre seréis, lo mejor de mi vida.

Live as if you were to die tomorrow.
Learn as if you were to live forever.
Mahatma Gandhi

Resumen

A lo largo de este proyecto, se ha diseñado, y desarrollado, una herramienta que permite el desarrollo y la distribución de tests, para la realización de medidas en Internet durante la fase inicial de pruebas de dichos tests, para facilitar, posteriormente, la implantación en una plataforma de producción.

El trabajo ha sido desarrollado dentro del marco del proyecto europeo Leone [LEONE], el cual tiene por objetivo la exploración de las soluciones actualmente existentes en los ámbitos de: gestión de red, plataformas de medidas globales, visualización de medidas, medidas en Internet y calidad de experiencia.

La herramienta desarrollada redundará en el beneficio de los diferentes participantes en el proyecto, facilitando el desarrollo de tests antes de su implantación y, por otro lado, constituye en sí misma una plataforma que se puede usar para obtener medidas en un entorno de investigación controlado.

Palabras clave: J2EE, Java, testbed, medidas en Internet a gran escala, MySQL, Web, plataformas de medidas, IP, Jitter.

Índice general

1. MOTIVACIÓN Y OBJETIVOS 7

1.1 Motivación 7

1.2 Objetivos 9

1.3 Estructura de la memoria 10

2. ESTADO DEL ARTE 12

2.1 Introducción 12

2.2 Planteamiento del problema 13

2.2.1 Herramientas para capturar y analizar tráfico y flujos de datos 14

2.2.2 Herramientas de muestreo 15

2.2.3 Herramientas de medidas para DNS 16

2.2.4 Herramientas de medidas para Web 17

2.2.5 Herramientas de medidas para P2P 18

2.2.6 Herramientas de medidas para juegos online 18

2.2.7 Otros 19

2.3 Análisis del estado del arte 20

2.3.1 RIPE 20

2.3.2 High-Energy Physics *22*
2.3.3 CAIDA *23*
2.3.4 PlanetLab *25*
2.4 Requisitos y restricciones 26
2.5 Marco regulador 28
3. DISEÑO Y DESARROLLO DE LA SOLUCIÓN 34
3.1 Introducción 34
3.2 Diseño 35
3.2.1 Elementos del testbed *36*
3.2.2 Casos de uso *37*
3.3 Desarrollo 42
3.3.1 Java *43*
3.3.2 Servidor: Apache Tomcat *43*
3.3.3 Base de Datos: MySQL *45*
3.3.4 Arquitectura: J2EE *46*
3.3.5 Conexión SSH: SSHJ *47*
3.3.6 Calendario: fullCalendar *48*
3.3.7 Herramientas de desarrollo *48*
3.3.8 Plataforma medidas de Samknows *50*
3.3.9 Caso de uso 1 – Insertar un nuevo test *52*
3.3.10 Caso de uso 2 – Planificación de tests *53*
3.3.11 Caso de uso 3 – Borrar un test existente *54*
4. PRUEBAS Y RESULTADOS 56
4.1 Introducción 56
4.2 Pruebas de unidad 57
4.2.1 Medida de la unidad de prueba nº1 ..... *58*
4.2.2 Medida de la unidad de prueba nº2 ..... *58*
4.2.3 Medida de la unidad de prueba nº3 ..... *59*
4.2.4 Medida de la unidad de prueba nº4 ..... *59*
4.3 Pruebas de integración 60
4.4 Pruebas de validación 61
4.5 Pruebas del sistema 63

5. PLANIFICACIÓN Y PRESUPUESTO 65
5.1 Introducción 65
5.2 Planificación 66
5.2.1 Planificación del proyecto 66
5.2.2 Diagrama de Gantt 66
5.3 Presupuesto 67
5.3.1 Costes de material 67
5.3.2 Costes de personal 69
5.3.3 Presupuesto total 70
6. CONCLUSIONES 72
6.1 Conclusiones 72
6.2 Posibles líneas de trabajo futuro 73
6.2.1 Instalación de tests 73
6.2.2 Ejecución de tests 75
6.2.3 Eliminación de tests 75
6.2.4 Planificación 75
6.2.5 Resultados 76
6.2.6 Gestión de usuarios 76
7. GLOSARIO 77
8. REFERENCIAS 80

Índice de figuras

Ilustración 1. Infraestructura del testbed 36
Ilustración 2. Diseño del caso de uso 1 37
Ilustración 3. Diseño del caso de uso 2 39
Ilustración 4. Diseño del caso de uso 3 40
Ilustración 5. Diseño del caso de uso 4 40
Ilustración 6. Diseño del caso de uso 5 41
Ilustración 7. Diseño del caso de uso 6 42
Ilustración 8. Sistema de directorios de Apache Tomcat 44
Ilustración 9. Diagrama E/R de la base de datos 45
Ilustración 10. Módulos de la aplicación web 46
Ilustración 11. Ejemplo de SSHJ 48
Ilustración 12. Ejemplo de fullCalendar 48
Ilustración 13. Panel de control de XAMPP 49
Ilustración 14. Eclipse 49
Ilustración 15. Plataforma de medidas de Samknows 50

Ilustración 16. Prueba de integración - Diagrama de clases 60
Ilustración 17. Prueba de integración – Diagrama de secuencia de caso exitoso 61
Ilustración 18. Planificación del proyecto 66
Ilustración 19. Diagrama de Gantt 67

Índice de tablas

Tabla 1. Prueba unitaria nº1 58
Tabla 2. Prueba unitaria nº2 58
Tabla 3. Prueba unitaria nº3 59
Tabla 4. Prueba unitaria nº4 59
Tabla 5. Prueba de validación – Estados 61
Tabla 6. Prueba de validación – Valores de entrada 62
Tabla 7. Prueba de validación – Casos de prueba 63
Tabla 8. Prueba de sistema 64
Tabla 9. Costes amortizables 68
Tabla 10. Costes no amortizables 68
Tabla 11. Coste total de material 69
Tabla 12. Cómputo de horas dedicadas al proyecto 69
Tabla 13. Honorarios 70
Tabla 14. Coste total de personal 70
Tabla 15. Coste final del proyecto 70

Capítulo 1

Motivación y objetivos

1.1 Motivación

Internet se ha convertido en una parte fundamental de nuestra vida. Si un empleado quiere comunicarse con su responsable simplemente le envía un correo electrónico con toda la información, si una persona desea comunicarse con un familiar que viva en el extranjero puede llamarle a través de Internet, etc. ¿Cómo hemos llegado hasta este punto?

"Internet es una red informática mundial, descentralizada, formada por la conexión directa entre ordenadores mediante un protocolo especial de comunicación" [RAE01] o dicho de una manera más sencilla "*Internet es una red de comunicación que conecta dispositivos digitales"*[CK06].

Actualmente casi cualquier dispositivo electrónico tiene acceso a Internet: ordenadores, teléfonos inalámbricos, tablets, naves espaciales, televisiones, videoconsolas, etc. Servicios de comunicación tradicionales como la televisión, la radio, la música o el teléfono están siendo redefinidos dando lugar a nuevos servicios como *Voice over Internet Protocol* (VoIP) [RFC4353]. Los usos que se le pueden dar a Internet son casi ilimitados.

Internet surgió de un proyecto de investigación, en los años 1960, sobre conmutación de paquetes. Los dos primeros nodos de Internet se implantaron el 29 de Octubre de 1969 en la primera red operacional de conmutación de paquetes *Advanced Research Projects Agency Network* (ARPANET) [Gro95]. A lo largo de estos más de 50 años, la red ha evolucionado hasta convertirse en una red desplegada mundialmente, con cientos de millones de usuarios y

dispositivos conectados [IWS]. La cantidad de información intercambiada actualmente está alrededor de los 2.5 trillones de bytes. Está previsto que la red siga aumentando en todos los aspectos posibles.

Uno de los mayores problemas que obtenemos del estudio de la red de Internet es, desafortunadamente, su mala caracterización a nivel global. Es precisamente esta mala caracterización la principal motivación del trabajo, como veremos a continuación. Esta situación es el resultado de tres factores condicionantes: descentralización, dinamismo de la red y factores técnicos y sociales.

En primer lugar nos encontramos que la red física de Internet no tiene nada que ver con su topología virtual. La principal causa de dicha descentralización es debido a que no hubo un diseño único sino que Internet surgió de diseños e implementaciones de organizaciones independientes, cada una de ellas con sus objetivos individuales. Esto dificulta cualquier caracterización que se quiera realizar en algún rango del mismo.

Por otro lado la red no se corresponde con un modelo estático sino que está continuamente cambiando; por cada segundo que pasa hay dispositivos conectándose y desconectándose en alguna parte de la red. El cambio se produce en todos los aspectos de la misma, tanto en tamaño como en configuración y tráfico. Las dificultades que esto representa a la hora de poder realizar las observaciones son claras: las medidas tomadas en un determinado lugar y un determinado tiempo no tienen por qué ser válidas para otro lugar o tiempo.

Por último, actualmente existen organismos que velan por la seguridad de cada ciudadano y que controlan qué datos son de carácter público y privado. Uno de los principales problemas a la hora de analizar el tráfico de datos medidas en Internet es salvaguardar esa privacidad y mantener siempre la toma de medidas dentro de los márgenes de la regulación vigente. Por otro lado los operadores no suelen ser muy cooperativos a la hora de proporcionar su jerarquía interna de conexión.

Además, con el transcurso de los años, la red se ha globalizado y aumentado de tal manera que la toma y el análisis de datos no pueden ser llevados a cabo con los métodos estadísticos estándar, dificultando enormemente la investigación en este campo. Se plantean así bastantes preguntas básicas sin respuesta:

- ¿Cómo de grande es Internet?
- ¿Cuál es la estructura de Internet?
- ¿Cuánto tráfico peer to peer está fluyendo en mi red?
- …

A pesar de todo lo comentado, a lo largo de todos estos años se ha avanzado mucho en la investigación acerca de la red. Se ha dedicado mucho capital y esfuerzo a intentar caracterizar protocolos, arquitectura y aplicaciones. Gracias a esta investigación las medidas en Internet cada día son más simples mejorando nuestra capacidad para evolucionar y avanzar.

Dichas medidas, cuya importancia es evidente, pueden ser clasificadas en función del interés de las mismas: comercial, social técnica.

En primer lugar, gracias a los resultados de las medidas de Internet es posible caracterizar y perfeccionar los productos y servicios que venderemos a los usuarios/clientes. Por ejemplo, un proveedor de Internet podría querer conocer la cantidad de usuarios que hay en una zona física, de cuánto ancho de banda disponen o qué tipo de tráfico suelen consumir.

En segundo lugar, cuanto más caracterizada se encuentre una red mejor podrán ser caracterizados sus usuarios. Actualmente hay compañías dedicadas por completo a la recopilación de información pública de cada usuario para luego distribuirlo a intereses privados u organismos públicos.

Por último, el interés técnico es obvio y totalmente necesario pues gracias a todas las medidas obtenidas en Internet, no solo se llevará a cabo una mejora de la red actual sino que se podrá avanzar hacia futuros objetivos mucho más ambiciosos. Todos los dispositivos que componen la red de Internet se basan en las estadísticas y caracterizaciones de la misma. Además cabe destacar que cuanto más caracterizada esté la red mejor más fácil será prevenir problemas en la red o con cualquiera de sus componentes.

En los últimos años se han realizado grandes esfuerzos para la recolección de información así como al desarrollo de herramientas y plataformas que facilitasen tanto la toma de medidas como la caracterización de la red. Dentro de todos esos proyectos destacan *NLANR* [NLANR], *CAIDA* [CAIDA], *MAWI-WIDE* [MAWI] y sobre todo *M-Lab* [MLAB] que cubren un amplio alcance o, en un rango mucho más específico, proyectos como *Crawdad* [CRWDD] dedicado a la toma de medidas de redes inalámbricas. Fruto de todos estos grandes esfuerzos se han publicado numerosas revistas y artículos como se puede observar en conferencias como *IMC* [IMC] o *SIGMETRICS* [SIG].

1.2 Objetivos

El principal objetivo de este proyecto es el diseño e implementación de una plataforma de pruebas (micro-testbed) que permita la colaboración para la realización de medidas en Internet en el contexto de la fase inicial del proyecto europeo *Leone: From global measurements to local management* [LEONE]. Las funciones principales a desempeñar en la fase inicial de LEONE se centran en la exploración de las soluciones actualmente existentes, en los ámbitos de: gestión de red, plataformas de medidas globales, visualización de medidas, medidas en Internet y calidad de experiencia

Antes de proceder a realizar el diseño se han analizado las principales plataformas de medidas existentes así como las publicaciones de las asociaciones dedicadas a las medidas en Internet. Además se ha realizado un estudio exhaustivo del draft *Large-Scale Measurement of Broadband Performance: Use Cases, Architecture and Protocol Requeriments* donde se desarrolla un estudio minucioso acerca de cómo realizar medidas en internet (protocolos, elementos,…) [SJM12]. Más adelante se profundizará sobre el mismo.

Posteriormente se ha hecho un exhaustivo análisis de la plataforma existente en el proyecto Leone así como en toda su infraestructura de pruebas. Una vez recopilada toda la información se ha procedido a validar el diseño implementando el mismo y ejecutando pruebas sobre el mismo.

Este proyecto no intenta centrarse únicamente en los aspectos técnicos relevantes para el diseño e implantación de la plataforma sino que además incluye información sobre los inconvenientes, ventajas y dificultades encontradas a lo largo del mismo además de detalles del software apropiado para la realización de las diferentes pruebas.

El diseño se ha realizado dentro del contexto del proyecto europeo Leone, el cual se encuentra actualmente en su fase de pruebas, donde ya existía un desarrollo inicial del mismo. El principal problema encontrado en el primer prototipo es que no estaba orientado hacia la colaboración para el desarrollo y ejecución de los tests que realizan las medidas en Internet, sino que cada participante desarrollara e implementara cada test de manera individual. Esto tiene una serie de problemas y es que, como puede suponerse, se puede dar el caso de que dos entidades desarrollen dos tests muy parecidos que hagan prácticamente lo mismo. Esto, no solo supone una pérdida de tiempo, sino costes. El proyecto debe llevar una agenda y tiene una fecha límite para la etapa de desarrollo (posteriormente se pasará a la siguiente fase donde se realizarán las pruebas a lo largo de toda Europa) y por lo tanto se hace crítico el parámetro del tiempo.

A continuación, se desglosan los requisitos de funcionamiento que deberá cumplir la plataforma de pruebas. Se puede destacar que estos mismos requisitos pueden aplicarse a cualquier otra plataforma de pruebas:

- Debe ser una aplicación web que permita el acceso a los usuarios independientemente de su ubicación.
- Los usuarios podrán compartir tests desarrollados por cada uno.
- Los usuarios además podrán planificar tests distribuidos.
- Debe ser fácil de usar para evitar pérdida de tiempo intentando superar dicha barrera de entrada.

En la fase de pruebas del proyecto, se ha querido desarrollar una infraestructura independiente de la infraestructura real del proyecto Leone. Para esto se ha realizado en paralelo un testbed de la plataforma de pruebas sobre una arquitectura de red basada en el diseño e implementación de la infraestructura real.

Para finalizar, se han realizado pruebas exhaustivas de todo el sistema. Para ello, se ha descompuesto el sistema completo en componentes validando su funcionalidad, tanto de forma independiente, como su comportamiento una vez se han integrado todos en un solo sistema completo.

1.3 Estructura de la memoria

En el capítulo 2 se detallan los resultados de la investigación realizada acerca del estado actual de las medidas en Internet, profundizando en la importancia de las mismas y las diferentes herramientas implementadas.

En el capítulo 3 se explica el problema que se desea resolver con el desarrollo de este proyecto, justificando las decisiones de diseño tomadas y enumerando las tecnologías utilizadas en su implementación.

En el capítulo 4 se desarrolla el proceso de pruebas al que ha sido sometido el proyecto.

En el capítulo 5 se presenta la planificación llevada a cabo para la elaboración de todo el trabajo desarrollado y el desglose del presupuesto final del mismo.

Capítulo 2

Estado del arte

2.1 Introducción

A lo largo de este capítulo se realizará un análisis en profundidad de la importancia de las medidas en Internet y en las diferentes soluciones propuestas e implementadas para que éstas sean llevadas a cabo además de intentar plasmar los requisitos y restricciones que surgen del proyecto realizado y un estudio en profundidad del estándar que define las arquitecturas de medidas en Internet.

En el primer apartado se realiza un estudio de la importancia de las medidas en Internet y de las soluciones abordadas a la hora de realizar las mismas mediante herramientas independientes. Además de analizarlas se clasificarán para intentar afrontar un amplio espectro con los diferentes parámetros y características que puede llegar a tener Internet con la máxima profundidad y de la manera más concreta posible.

A continuación, se procederá a analizar los problemas que pueden surgir en las herramientas independientes así como la importancia que tienen las organizaciones y plataformas de medidas en Internet. Para ello, se toman como punto de partida tres organizaciones características y una de las plataformas más relevantes actualmente. A lo largo del análisis se observarán las ventajas y dificultades que surgen del trabajo continuo de estas entidades así como una visión global de su trabajo actual y pasado.

En el tercer apartado se intentarán abordar los diferentes requisitos y restricciones que existen sobre el proyecto desarrollado. Puesto que el proyecto se realiza en el marco del proyecto europeo [LEONE] muchas restricciones y requisitos vendrán impuestos por el mismo.

Por último se abordará el documento que define cómo debe implementarse una arquitectura de medidas en Internet [SJM12]. Para ello se intentará arrojar luz sobre diferentes aspectos de la arquitectura, como qué elementos debe tener, qué protocolos se debe seguir para la correcta interacción entre ellos, además de un completo listado de requisitos que debe cumplir para su correcto funcionamiento.

2.2 Planteamiento del problema

Internet se ha convertido en una infraestructura crítica, lo cual se refleja en el hecho de que muchos gobiernos están formulando políticas en la Sociedad de la Información en general y en Internet en particular [KBV+04]. Se puede considerar como un gran sistema distribuido que comprende el usuario al que conecta, el tráfico en forma de paquetes generados y los propios protocolos que gobiernan la transmisión de los paquetes entre usuarios. La variabilidad de las demandas de tráfico y la complejidad de las interacciones presentan un reto para los proveedores de internet, que deben asegurar que las fuentes responden adecuadamente a las demandas [Duf04].

Esta gran complejidad técnica y organizativa en la infraestructura de Internet dificulta encontrar cuellos de botella y puntos de fallo individuales, así como identificar dependencias críticas en ciertos ISPs [KVV06].

Hay una gran variedad de tráfico en la red: trazas de paquetes, tablas de rutas, logs, nombres DNS para direcciones IPs,… Además del tráfico conocido, hay una gran cantidad de datos que desconocemos. Lo cual nos hace plantearnos preguntas, por ejemplo, ¿cómo se determina la importancia de la investigación de los datos? [SM04]

Las medidas del tráfico se llevan a cabo por varias razones entre las que se incluyen la monitorización y el control de la red [Peu02]. Entre el conjunto de razones podemos distinguir tres bien diferenciadas: caracterización de comportamiento, problemas de diseño y manejo de la red. La primera nos puede dar medidas de rendimiento y nos ayudará a entender su crecimiento. La segunda nos ayudará a entender problemas que surjan, sugerir mejoras, entender problemas de protocolo y entender su comportamiento. Por último, el manejo de la red nos puede ayudar a diagnosticar, corregir y distinguir entre patrones de tráfico. [Bad]

Se puede destacar una razón por encima del resto: no se puede mejorar Internet si no se llega a entender su estructura y su comportamiento. Es decir, no podemos mejorar Internet ni construir modelos efectivos o simuladores si no realizamos medidas [Bar04].

En conclusión, las medidas en Internet no son solamente relevantes para ingenieros de redes e investigadores, sino también para analistas que buscan identificar, comprender y resolver problemas específicos en áreas de competitividad, fiabilidad y rendimiento. La única forma de obtener los datos necesarios para ese grado de penetración en la red es mediante medidas en internet, que permiten definir la topología de la misma [KVV06].

A lo largo de los años han sido desarrolladas un gran conjunto de herramientas para medir parámetros de Internet. A continuación se va a desglosar una clasificación de herramientas de medidas en Internet individuales [CK06].

2.2.1 Herramientas para capturar y analizar tráfico y flujos de datos

En primer lugar se encuentran los métodos activos de descubrimiento de propiedades de infraestructura. Cuando se habla de medidas activas se refiere a la inyección de tráfico con el propósito de toma de medidas. Dentro de estas herramientas se puede destacar:

- ***Ping*:** muy útil para comprobar la conectividad entre dos máquinas. Se trata de una de las herramientas más extendidas por su fácil manejo y su gran utilidad.
- ***Zing*:** se trata de una variedad de la herramienta *ping* que además permite planificar el envío de los RTT [ZDPS01].
- ***Traceroute*:** descubrimiento de red usando el campo de la cabecera IP, TTL. Un sistema de medida a gran escala que usa esta herramienta para descubrir la topología de la red es *skitter*.
- ***Scriptroute***: herramienta que permite a usuarios sin privilegios la habilidad de capturar paquetes de la red e inyectar paquetes en la red, con reglas establecidas por los administradores de sistema [SWA03].
- ***PeriScope***: librería programable implementada dentro del núcleo del sistema operativo, con una interfaz de aplicación programable (API). Las aplicaciones pueden definir nuevas estructuras de sondeo e inferir técnicas para extraer resultados del patrón de llegada de las respuestas de las sondas [HBB02].

En contraposición con las anteriores tenemos los métodos pasivos para descubrir propiedades de las infraestructuras. Los métodos pasivos se basan en la captura de tráfico generado por otros.

- **BGP:** el sistema de enrutamiento inter-dominio determina cuanto tráfico es intercambiado entre sistemas autónomos. La tabla de enrutamiento de BGP proporciona información parcial sobre la topología a nivel de sistema autónomo. El repositorio *routeviews* recolecta observaciones de BGP a partir de un gran conjunto de Ases [Rou] Este se ha convertido en un gran origen de datos muy útiles para monitorización y análisis de la topología de Internet pasiva.
- **OSPF:** consiste en la captura de LSAs en un dominio específico. Se puede obtener Detalles y ejemplos de análisis de OSPF en [SIG+02].

Como se puede observar hay dos corrientes bien distinguidas de cómo realizar la toma de medidas, aunque también se encuentra desarrollándose herramientas que combinan estas dos técnicas: pasivas y activas.

Dentro de todas las características sobre las que se puede realizar la toma de medidas se puede destacar dos: medidas de ancho de banda y medidas de latencia. Estas medidas en particular son muy útiles en diferentes ámbitos: aplicaciones streaming, selección de servidores, redes superpuestas, verificación de los acuerdos a nivel de servicio, etc.

Por último, se deben destacar las herramientas de geo-localización. La localización de los elementos de la red puede ser muy útil para una gran variedad de razones sociales, económicas y para muchísimos aspectos relacionados con ingeniería (como se puede evidenciar en la serie de talleres comenzados en 2003) [TS05]. El grupo de trabajo *geopriv* junto con el IETF han definido una serie de estándares para manejar la información de localización. El documento de requerimientos para la autorización, seguridad y privacidad en el manejo de información de localización es RFC 3693 [CMM+04].

2.2.2 Herramientas de muestreo

Estas herramientas se enfocan en la captura y el análisis del tráfico de la red. Las diferentes formas de captura de datos pueden clasificarse de la siguiente manera: la captura de paquetes en sistemas de interés general y en sistemas de interés especial.

Un sistema final puede típicamente capturar y guardar todos los paquetes que pasen por sus interfaces, y frecuentemente es lo más conveniente [Mog90]. Se pueden destacar:

- ***Libcap:*** librería que incluye puntos de entrada para especificar las interfaces que van a ser monitorizadas y los tipos de paquetes que van a ser guardados. Se basa en el uso de filtros.
- ***Pktd*** [GP03] y ***scriptroute*** [SWA03]: permiten la captura de paquetes de la red e inyectan paquetes en la misma con los filtros establecidos por los administradores de red.

La captura de paquetes de otros enlaces que no sea la red local (LAN) puede ser muy costosa aunque el tráfico capturado tiende a ser más representativo y diverso. Los problemas que nos encontramos son tecnología más variada y tasas del enlace mucho más rápidas. Generalmente necesitamos hardware especializado; por ejemplo, para enlaces ópticos, se puede utilizar un divisor óptico [AkcTW97, CJs03]. Algunos ejemplos de productos no comerciales de captura de paquetes son los sistemas ***OC3MON*** y ***OC12MON*** [Cor, TMW97].

Se puede obtener tráfico del plano de control directamente participando en la comunicación del plano de control (p.ej. intercambiando mensajes de enrutamiento). Por ejemplo, se puede configurar un ordenador para que hable BGP con ***GNU Zebra*** [gnu] o ***Quagga*** [quaa], luego traducir los binarios usando el kit de herramientas ***MRTD*** (en particular ***route_btoa***) [MRTa]. Un gran repositorio de tráfico BGP se encuentra archivado en el proyecto ***Routeviews*** [Rou].

Por otro lado, existen herramientas especializadas en el manejo de datos; es decir, en la captura de datos y su almacenamiento:

- ***Sistema smacq***: conjunto de módulos para el procesamiento de captura de datos [FV02].
- ***Windmill***: sistema enfocado en la medida de rendimiento de los protocolos [WMJ04].
- ***Nprobe***: similar a ***Windmill*** pero incorporando explícitamente reducción de datos online para proveer escalabilidad [MHK+03].
- ***Sistema FLAME***: provee procesamiento extensivo dentro del núcleo a través del uso de extensiones seguras del propio núcleo [AIM+02].

- ***Coral Reef***: sistema organizado como un conjunto de herramientas para leer y procesar tráfico de datos en una variedad de formatos [KMK+01].
- ***Sistema STREAMS*** [ABB+05] y ***TelegraphCQ*** [KCC+03, PGB+04]: bases de datos de flujos de tráfico utilizando SQL limitado.
- ***Gigascope:*** se trata de un sistema de manejo de flujos de datos especializado en el análisis de tráfico y usado en conjunción con captura de paquetes de alto rendimiento [CJSS03b, CJSS03a].

Un problema a la hora de la toma de las medidas es la gran cantidad de datos que se obtienen. Para manejarlos, se utilizan diferentes métodos. Los más comunes son:

- Contadores: la manera más común de hacer un resumen del tráfico es utilizar la agregación para formar series de tiempo de conteos de estadísticas del tráfico, por ejemplo, bytes o paquetes por unidad de tiempo. Esto se puede conseguir en todos los routers vía el ***MIB-II Management Information Base***, el cual es accesible vía SNMP [MR]. En una Ethernet se pueden obtener estadísticos de ***RMON Management Information Base*** [Wal00]. Estos estadísticos son llamados conteos de SNMP y se trata del único MIB que ha conseguido status de IETF estándar y todos los agentes de SNMP lo usan.
- Mientras que los contadores proveen información básica acerca de la actividad de la red, toda la semántica del tráfico queda ausente. Una mejor manera es la captura de flujos de tráfico de datos (centrado en números de bytes o paquetes que llevan un valor particular 5-tuple en un intervalo) o almacenar trenes de paquetes (el cual puede ser usado para monitorización de actividad básica de la red, monitorización de los usuarios y aplicaciones, planificación de la red, análisis de seguridad y contabilidad y facturación (detalles en [Cis])). El conjunto de herramientas ***Flowtools*** puede guardar, enviar y procesar y generar reportes de Cisco NetFlow o Juniper cflow [Ful].
- Aunque estas dos sean las más habituales, existen otros métodos como el muestreo.

2.2.3 Herramientas de medidas para DNS

Hay múltiples propiedades de la aplicación DNS que puede ser de interés para medir. Entre ellas podemos destacar:

- Fracción del tráfico de Internet
- Disponibilidad: relevante para la infraestructura. Se mide en los servidores raíz.
- Número de entidades: útil para medidas de rendimiento. Suele medirse a través de ingeniería inversa a distancia.
- Latencia de la respuesta: sirve para medidas de rendimiento. Se mide sobre un conjunto de servidores.

La herramienta más importante y más extendida es ***dig*** (Domaing Internet Groper) y es usada para buscar información en la base de datos global de DNS. Aparte de esta herramienta podemos encontrar muchísimas más, las cuales se pueden clasificar en medidas pasivas, medidas activas y monitorización activa.

Dentro de las medidas pasivas nos encontramos con:

- Netflow logs: caracterización local del tráfico DNS.
- Logs de actualización de DNS: caracterización de tráfico y autenticación.
- Representación en grafo: clasificación de entidades DNS.
- ***NeTraMet***: espectroscopia de DNS [Nete].

Dentro de las herramientas que usan monitorización activa tenemos:

- ***dnsstat***: obtención de estadísticas DNS locales [Dnsf].
- ***dnstop***: variante de dnsstat pero focalizado en eventos inusuales [Dnsg].
- ***dsc***: extensión de dnstop utilizada para la obtención de estadísticas DNS locales filtradas para ayudar a resolver problemas [Dnsd].

Por último respecto a las herramientas con medidas activas:

- ***fpdns***: identificar implementación DNS [AS].
- ***dnschecker***: identificar nodos en la resolución de ruta DNS [DNSc].

2.2.4 Herramientas de medidas para Web

Las propiedades Web de interés para ser medidas pueden ser clasificadas por clases:

- Caracterización a alto nivel: fracción de tráfico y número de entidades. Sirve para examinar las tendencias generales.
- Localización: presencia de las entidades Web. útil para manejar distribución de la población y su movilidad.
- Configuración: configuración software/hardware. Utilizado para habilitar manejo de carga.
- Modelos de trabajo de usuario: patrón de acceso. gracias a esta propiedad se puede modelar el fenómeno Web y el cambio de las poblaciones de usuario.
- Propiedades de tráfico: almacenamiento en caché. Sirve para aprovisionamiento de condiciones regulares y anormales.
- Demanda de aplicaciones: impacto en la red. Utilizado para la mejora de protocolo.
- Rendimiento: rendimiento de los componentes Web. Sirve para mantener la popularidad.

Dentro de todas las herramientas para web destacamos:

- Herramientas de caracterización
- Medidas de rendimiento: Hay una gran variedad incluyendo ***Keynote*** [Keya], ***Akamai*** [Aka], ***eValid Test Suite*** [Eva] y ***Gomez*** [Gom]. Las compañías además usan productos para chequear que si se cumplen los protocolos p.ej. ***Co-Advisor*** [COA].

- Network-aware clustering: ha sido mostrado como una técnica efectiva para agrupar direcciones IP que estén cerca topológicamente y bajo algún tipo de control administrativo [KW00].
- Para el manejo de clientes móviles: Intermediarios especializados [SP02] han sido propuestos para personalizar el torrente de flujo de los clientes móviles.

2.2.5 Herramientas de medidas para P2P

Desde la introducción de ***Napster*** [Nap] en 1999 la fracción del tráfico de Internet dedicado a Peer-to-Peer ha incrementado de forma inimaginable. Actualmente ha cobrado mayor importancia el protocolo ***BitTorrent*** [Bit].

Dentro de las propiedades del tráfico P2P destacamos:

- Fracción de tráfico de Internet: relevante para medir patrones de crecimiento.
- División del protocolo: seguimiento de las tendencias en el contenido de intercambio.
- Entidades de localización: útil para el rendimiento y la agrupación
- Maneras de acceso: usada para mejorar la eficiencia en la búsqueda y la descarga.
- Latencia de respuesta: útil para obtener medidas de rendimiento.
- Disponibilidad de los nodos: utilizado para el rendimiento y el alcance.

Hay bastantes herramientas de P2P que se pueden clasificar fácilmente en: herramientas de caracterización, de examinación de la arquitectura, herramientas para hacer frente a los aspectos únicos de P2P y herramientas para dar soporte a nivel de red.

2.2.6 Herramientas de medidas para juegos online

El mundo de los juegos online ha crecido a lo largo de los últimos años y ha conseguido hacerse un hueco entre el tráfico intercambiado a nivel mundial. Dentro de todas las propiedades que pueden ser interesantes para medir destacamos:

- Caracterización del tráfico: sirve para medir la popularidad y los patrones de crecimiento.
- Arquitectura del sistema de juego: diferenciar la arquitectura.
- Escalabilidad: útil para el rendimiento y aprovisionamiento.
- Requerimientos de tiempo real: limitaciones de latencia y viabilidad del juego.
- Maneras de acceso: restricciones de movilidad
- Duración de la sesión: mantenimiento del estado.

Entre todas las herramientas destacamos:

- ***qstat***: despliega estado del servidor de diferentes juegos, información de los jugadores y otras estadísticas para la variedad de juegos FPS [Qst].
- ***GameSpy*** [Gam] y ***Steam*** [Ste]: son aplicaciones que han sido creadas para informar a los jugadores de un número de jugadores asociados con ese servidor de juego.

2.2.7 Otros

- ***RealTracer***: herramienta para la medición del rendimiento de transmisión basado en encuestas [WCZ01].
- ***mmdump***: herramienta de monitorización para facilitar la captura y procesamiento de tráfico multimedia [vdMCCS00].
- Además cabe destacar la gran relevancia de las herramientas especializadas en anonimizar los datos y en seguridad de los cuales profundizaremos más adelante.

A pesar de lo anterior el conjunto de herramientas en el campo de las medidas de internet no está tan extendido como en otras áreas de la ciencia de la computación. Entre todas las herramientas comentadas destacamos cuatro diferentes conjuntos de herramientas que representan diferentes puntos a lo largo del espectro software:

- ***Windmill*** [MJ98a, WMJ04]
  - Principalmente se trata de una herramienta de medición pasiva dirigida al rendimiento de protocolos aunque puede ser usado con herramientas de pruebas activas sin aplicar modificaciones.
  - Tiene tres componentes funcionales: filtrado por protocolo, módulos abstractos de protocolo y un motor extensible basado en un cargador dinámico. IP, TCP, BGP, HTTP, etc.
- ***Click***
  - El router Click permite montar configuraciones complejas partiendo de partes simples que llevan a cabo funciones de router específicos como clasificar paquetes, planificación, etc. Mediante la apertura de las funciones de un router y mapeo de módulos (llamados elementos) para cada una de las funciones principales, Click permite configuraciones arbitrarias complejas del router.
  - Aunque Click nació para hacer encaminamiento de paquetes, ha sido usado en medidas de Internet [Koh06] y entornos relacionados. Teniendo en cuenta que un número significativo de medidas es requerido para el manejo de paquetes y análisis de trazas, es natural utilizar Click para diversas tareas de medición de paquetes. Puede ayudar en reconstruir corrientes de paquetes.
  - La herramienta ***ipsumdump*** [Koh] utiliza Click para varias actividades que van desde leer hasta escribir ficheros tcpdump a anonimizar datos.

- ***Dss*** [FK05]
  - Herramienta Unix útil para el escaneo de flujos de datos que puede llegar a procesar gigabytes de datos.
  - Provee una abstracción a los datos para reducir el parseo de diferencias entre varias herramientas de análisis, con un soporte eficiente I/O en espacio y tiempo y una interfaz uniforme para todos los datos.

  - Una aplicación fue el proyecto monitor OSPF [SG01] donde datos LSA eran guardados de la red y estos eran seleccionados basados en expresiones de petición especificados por el usuario, para procesamiento de subsecuencias y análisis.
- ***Gigascope*** [JMSS05, CJSS03b]
  - Es una combinación de hardware adaptado y software de base de datos que han sido usados en captura de datos pasivos en altas velocidades y procesamiento activo de consultas permanentes especificadas y unidades especificadas por el usuario.
  - Se trata de un sistema de gestión de flujo de datos dirigido a monitorizar los flujos de datos a altas velocidades.
  - Ha sido utilizado en una amplia variedad de aplicaciones tales como el análisis de tráfico, detección de intrusos y análisis de protocolos.

El problema que surge de las herramientas encargadas de esnifar de tráfico de alta velocidad es que tienen un conjunto rígido de las tareas de supervisión relacionadas con ellos. Esto dificulta expandir su funcionalidad más adelante cuando surjan nuevas necesidades.

Además los tests citados se realizan de forma independiente y puede resultar muy tedioso a la vez que conflictivo, lo cual hace necesario plataformas de medidas que puedan usar los proveedores, etc., más allá de los tests sueltos.

2.3 Análisis del estado del arte

A lo largo del capítulo pasado se ha profundizado en un amplio rango de herramientas de medidas en Internet individuales. El uso de todas estas herramientas de forma individual e independiente da lugar a varios problemas; por ejemplo, el que los resultados ofrecidos por la toma de medidas en Internet en cada una no se pueden generalizar a toda la red de Internet. Por esta razón, surgen proyectos a gran escala y plataformas de medidas. A continuación se profundizará en tres organizaciones que realizan proyectos de medidas a gran escala (*RIPE*, *High Energy Physics* y *CAIDA*) y en una de las plataformas de medidas más importantes actualmente, *PlanetLab*.

2.3.1 RIPE

Se trata de un foro abierto con un interés específico en el desarrollo técnico de Internet. *RIPE* no tiene miembros fijos, cualquier persona que esté interesado en el trabajo de esta organización puede participar a través de listas de correos y asistiendo a reuniones. La comunidad *RIPE* se comunica a través de listas de correos electrónicos [RIPEm], grupos de trabajo [RIPEwg], y reuniones [RIPEme].

RIPE obtiene su nombre del francés "*Réseaux IP Européens*", que traducido al castellano quiere decir "*Redes IP Europeas*" y no se trata de un organismo de normalización como el [IETF] ni se ocupa de nombre de dominio como [ICANN].

RIPE NCC (*Réseaux IP Européens Network Coordination Centre*) es una entidad no lucrativa establecida en 1992 y financiada por sus miembros. La función principal de esta entidad es proveer soporte administrativo a *RIPE* y a los grupos de trabajo, así como la de facilitar reuniones. Además cabe destacar que se trata del *RIR* (*Regional Internet Registry*) para Europa, Oriente Medio y parte central de Asia [RIPE-NCC].

Una de las tareas principales del *RIPE NCC* es la asignación de espacios de direcciones IP y números de AS y el mantenimiento de la base de datos de Whois a lo largo de la región *RIPE* entre otras. Además *RIPE NCC* cumple con varias funciones entre las que podemos destacar ser el coordinador para varios puntos de intercambio y operar uno de los trece servidores raíz DNS, k.root-servers.net [RS].

RIPE lleva a cabo una gran variedad de proyectos relacionados con medidas de Internet. A continuación, se explicará el resultado del estudio de tres proyectos representativos de las diferentes áreas en las RIPE que desempeña sus proyectos.

En primer lugar se tiene el área relacionado con el crecimiento y cambio de Internet en el cual se puede destacar el proyecto *Hostcount* [Hostc]. Uno de los principales objetivos en este área en general y en este proyecto en partículas es el de proveer información a la comunidad de Internet de cómo está creciendo la red para lo cual *RIPE* lleva contando desde su inicio cuántos hosts están registrados en el DNS a lo largo de la región *RIPE*. Este conteo es llevado a cabo cada mes y los resultados son publicados para que sean disponibles para toda la comunidad de Internet.

En segundo lugar se tiene los proyectos relacionados con la interacción del tráfico y la red dentro del cual se puede citar el proyecto [TTM]. Se trata de un proyecto a gran escala que comienza en el año 2000 y que tiene por objetivo proveer resultados de medidas activas relacionadas con la conexión a Internet, medios para diagnosticar problemas de rendimiento y medidas de las tendencias a largo plazo en la conectividad de la red.

En concreto, el proyecto *TTM*, mide retrasos en el tráfico de una sola dirección entre hosts (latencia), pérdida de paquetes, información del camino (traceroute) y variación del retraso (jitter). Su infraestructura se basa en dos componentes: sistemas de medida *TTM* que toman medidas periódicamente con el resto de sistemas y un centro de operaciones en *RIPE* que almacena los resultados de estas medidas garantizando seguridad.

Uno de los aspectos más interesantes del proyecto *TTM* es que se trata de uno de los proyectos a gran escala con sincronización entre sus componentes más grandes implementados. Los sistemas *TTM* toman una gran variedad de medidas: tráfico, red, topología y ancho de banda, cumpliendo con los estándares de *IPPM* para medidas activas de la red [PAMM98]. Por último destacar que es de uso fácil: el usuario conecta el sistema a su red y el propio personal de *RIPE* es el encargado de mantener y configurar todo el software.

Por último se va a analizar el tercer área sobre la que trabaja *RIPE*: medidas pasivas de BGP, en concreto se va a focalizar sobre el proyecto *Routing Information Service* [RIS]. El proyecto *RIS* comienza en 1999 y su objetivo principal es ayudar a los operadores de la red a resolver problemas en enrutamiento ofreciendo datos obtenidos tanto de la captura de paquetes *BGP* como de las tablas de enrutamiento *BGP* a lo largo de la región *RIPE*. La

captura se realiza usando el suite de enrutamiento *Quagga* [quaa]. Además cobra mayor importancia debido a que soporta una gran variedad de consultas como, por ejemplo, cuándo ha aparecido un prefijo por última vez en la tabla de enrutamiento global.

En 2010 y como continuación del proyecto TTM surge el proyecto RIPE Atlas. Con este proyecto, RIPE NCC persigue el objetivo de construir la infraestructura de medidas más grande de todo el mundo gracias a la ayuda de las personas que quieran participar. La infraestructura consiste en una red global de probes que miden conectividad y accesibilidad. RIPE NCC recolecta los datos resultantes y pone a disposición de la comunidad de Internet en forma de gráficos [Atlg] o de mapas [Atlma]. Los usuarios de RIPE Atlas que actúan como host para las probes pueden elaborar y lanzar sus propias medidas personalizadas a lo largo de la red RIPE Atlas.

Los objetivos principales de este proyecto son: proveer medidas activas a lo largo de miles de puntos de observación de Internet para usuarios, producir mapas de carácter técnico de la red de Internet y actuar como fuente confiable de información sobre la vida real a través de medidas activas.

La comunidad RIPE Atlas está constituida por usuarios (cualquiera que acceda a los mapas y estadísticas de RIPE Atlas), hosts (cualquiera que conecte la probe a su red hogar), sponsors (aquellos que proveen soporte financiero para un número de probes) y miembros de RIPE NCC que adquieren una serie de ventajas sobre la infraestructura.

El funcionamiento consiste en que usuarios sirvan como hosts a las probes conectándolas a su red hogar. Los dispositivos realizan medidas (ping, traceroute, DNS y SSLcert), consumiendo el mínimo ancho de banda necesario, y envían los resultados al repositorio central de RIPE NCC.

Como resultado se obtienen multitud de beneficios para usuarios, hosts, sponsors y miembros de RIPE NCC. Por un lado, tanto los hosts como los sponsors pueden ganar créditos que les permitan lanzar sus propias medidas personalizadas [Atludm]. Por otro lado, los miembros de RIPE NCC, aun no sirviendo como hosts para las probes, tienen diferentes ventajas añadidas como, por ejemplo, probar su accesibilidad IPv6 [Atlme]. Por último, la comunidad de Internet puede acceder a los datos de RIPE Atlas en los diferentes formatos disponibles y usarlos para su propio propósito como, por ejemplo, realizar un caso de estudio del filtrado IPv6 /48 [Atlif] o realizar una comparación de tiempos de respuesta de servidores DNS [Atlcut] entre otros. [Atlauc]

Toda la información para hacerse partícipe de este gran proyecto se puede encontrar en [Atlgi].

2.3.2 High-Energy Physics

La comunidad de física de alta energía [HEP] lleva a cabo operaciones de medidas de Internet focalizadas en el rendimiento de Internet y la reducción de la latencia.

Están organizados por el comité *International Committe for Future Accelerator* [ICFA], el cual tiene un *Standing Committe on Interregional Connectivity* [*SCIC*]. El *SCIC* se encarga de la monitorización inter-regional de la conectividad de la red de Internet además de

rastrear requisitos de los servicios de la red de la comunidad *HEP* y sugerir recomendaciones para la mejora de la infraestructura de la red.

A continuación se detallan tres proyectos llevados a cabo por el grupo de trabajo *Monitoring Working Group* dentro del *ICFA-SCIC*. Este grupo de trabajo se encuentra localizado en el centro de *Stanford Linear Accelerator Center* y su objetivo principal es el de proveer una visión del rendimiento de forma cuantitativa y técnica inter-regional de la red para permitir el conocimiento de la situación actual y sugerir recomendaciones para la mejora de la conectividad inter-regional [Cot05a].

En primer lugar se encuentra el proyecto *pingER*, que lleva operando desde 1995. El objetivo principal es obtener datos de rendimiento y tendencias a lo largo de la red enviando pings a sondas y guardando resúmenes de las estadísticas obtenidas para que luego puedan ser vistas por instituciones educacionales y de investigación [Cot05b]. Para alcanzar dicho objetivo se ha desplegado una infraestructura de medidas de las más grandes existentes. Gracias a la recolección de esta gran cantidad de datos se puede llegar a tener una idea de cómo ha cambiado la red a lo largo de todos estos años, por ejemplo mirando la reducción de las tasas de pérdidas de los paquetes o la variación de la latencia en los caminos de *HEP*.

Mientras que, como se ha podido observar, *pingER* hace énfasis en las medidas de latencia y de tasas de pérdidas de paquetes, los investigadores necesitan un alto rendimiento para sus aplicaciones, lo cual motiva el nacimiento del proyecto [IEPM-BW]. Este proyecto usa la herramienta [Iperf] para medir el rendimiento de los caminos intentando enviar a la máxima tasa posible cada diez segundos por cada uno [TCD03]. Gracias a este proyecto podemos obtener medidas muy interesantes de cara al rendimiento de los caminos pero también da lugar a una serie de problemas, ya que un test sencillo puede llegar a consumir cientos de megabits por segundo.

Las actividades de monitorización de la red de *ICFA-SCIC*, que en un principio se centraban en soporte de *HEP*, se han terminado desplazando hacia la caracterización y medida de la brecha digital. Este término se refiere a las diferencias en términos de computación e infraestructura de red entre las diferentes regiones del mundo.

2.3.3 CAIDA

Se trata de una organización sin ánimo de lucro con soporte y participación de diferentes sectores: comerciales, gubernamentales y de investigación. La organización fue fundada en el año 1997 y desde entonces se han realizado multitud de proyectos de medidas de Internet y análisis de datos de Internet.

Los objetivos de *CAIDA* son varios, entre los que destacan proporcionar una mayor comprensión de la infraestructura de Internet y proveer acceso al análisis del tráfico y a herramientas de visualización para simplificar la administración y la toma de medidas de la red.

Su trabajo se centra principalmente en tres áreas: construir herramientas para las medidas de Internet (tanto software como hardware), recolectar datos y medidas de Internet y realizar un análisis de éstas últimas.

Respecto al desarrollo de herramientas, *CAIDA* ha realizado un gran esfuerzo en el desarrollo de las mismas para facilitar y mejorar la toma de medidas en la red así como para recolectar y analizar tráfico en términos de flujo. Algunos ejemplos de herramientas soportados por *CAIDA* son:

- ***Iffinder*** [Keyb]: realiza resolución de alias y mapeado de interfaces IP a routers.
- ***Gtrace*** [PN99]: provee interfaz geográfica a la herramienta *traceroute*.
- ***NeTraMet*** [Nete]: implementa el estándar *RTFM* para las medidas de flujos y contabilidad.
- ***Skitter*** [HPMkc02]: proyecto a gran escala enfocada al descubrimiento de topología.

CAIDA también tiene un rango de proyectos relacionados con la recolecta de datos. La organización lleva recolectando datos desde el año 2002, los cuales están disponibles aplicando sobre ellos previamente técnicas de anonimización para garantizar la privacidad de los usuarios.

Una forma novedosa es la manera que tiene de recolectar los datos. *CAIDA* junto con *UCSD* han formado la red telescopio, que consiste en un conjunto particular del espacio de direcciones IPv4, abarcando la mayoría del prefijo /8, el cual no es usado para la conectividad de la red estándar; eso hace que el tráfico entrante deba ser anómalo en algún sentido.

El uso más común a lo largo de la red telescopio es el de análisis de seguridad. Por ejemplo, los gusanos normalmente escanean espacios de direcciones para buscar hosts a los que infectar; ese tráfico de escaneo tiende a llegar a la red telescopio.

Un proyecto de medidas activas a gran escala es el proyecto *Skitter*. Este proyecto consiste en la recolecta de datos de la topología de la red utilizando un mecanismo parecido al de la herramienta *traceroute*. El principal objetivo del proyecto *Skitter* es proveer datos de la topología de la red para el análisis de los caminos en Internet. Este proyecto ha sido útil para arrojar luz sobre herramientas de medidas, métodos de estimación de latencia, métodos de geo-localización, propiedades de grafos y propiedades de muestreos de grafos.

Tras diez años sirviendo a la comunidad de investigadores, el proyecto Skitter ha evolucionado en el nuevo proyecto Archipelago [Ark]. Este nuevo proyecto es la infraestructura de medidas activas de próxima generación de CAIDA, y tiene como objetivos reducir esfuerzo para el desarrollo y despliegue de medidas a gran escala y permitir a los colaboradores ejecutar sus medidas en una plataforma distribuida con seguridad garantizada [CMC05].

Ark es una infraestructura diseñada para correr sobre ella medidas activas, facilitando así su diseño e implementación. Esto permite que los desarrolladores enfoquen sus esfuerzos en reducir la dificultad y mejorar la comunicación para la realización de medidas distribuidas. Por ejemplo, para la realización de medidas que requieran una sincronización precisa han desplegado RADclock a lo largo de los monitores *Ark* [RADc].

El objetivo principal de *Ark* es la coordinación de las medidas de topología a gran escala basadas en traceroute a través del proceso llamado *team probing*, a través del cual se agrupan monitores en equipos y se divide el trabajo de las medidas de forma dinámica entre sus miembros. Este proceso permite ahorrar tiempo y obtener resultados más acertados.

En *Ark* las medidas son tomadas usando la herramienta *scamper* [Sca]. Los resultados de las medidas son distribuidas a través del IPv4 Routed /24 Topology Dataset [TDa]. Además**,** a través del servicio de búsqueda DNS que han diseñado llamado *HostDB,* realizan lookups automatizados de todas las direcciones del dataset anteriormente citado.

Ark además recoge datos de topología IPv6 [TDb] de los monitores Ark, y almacenan en un dataset los enlaces entre Sistemas Autónomos derivados de los paths de los datos almacenados en los dataset de topología, lo cual es útil para el estudio de las relaciones entre los proveedores de Internet.

Parte de los nodos de la infraestructura de Ark están participando en el proyecto Spoofer el cual recoge tráfico potencialmente spoofed y lo reencamina hacia un servidor en el MIT que realiza el análisis del mismo [Spoof].

Por último, un ejemplo del análisis de datos realizados por *CAIDA* es el que concierne al emplazamiento de los servidores raíz DNS. El rendimiento de estos servidores es crítico ya que una gran cantidad aplicaciones dependen de él, incluyendo Web. A pesar de esto, el emplazamiento de los servidores no fue escogido de forma óptima, sino por razones históricas. Para investigar la cuestión citada, CAIDA instaló monitores *Skitter* en la mayoría de las localizaciones de los servidores raíz. El siguiente paso fue inferir en qué servidor raíz esperaría visitar ese cliente, lo cual permite una medida directa del rendimiento.

2.3.4 PlanetLab

PlanetLab surge en el año 2002 [PACR02] como una plataforma global para el desarrollo y testeo de servicios en la red. Su diseño se basó en la introducción de nuevos servicios a la red de Internet. PlanetLab está formado por un gran conjunto de nodos desplegados a lo largo de la red sobre los que corren máquinas virtuales que monitorizan el software del nodo, planifican los recursos del nodo y cumplen funciones de seguridad (por ejemplo protegiendo a los nodos de ataques maliciosos).

Los usuarios pueden acceder a los nodos a través de la capa de abstracción *slice*. La plataforma *PlanetLab* es usada en multitud de proyectos, no todos relacionados con las medidas de Internet aunque sí en su mayoría.

Existen estudios que demuestran que tomar el conjunto de nodos de la plataforma de medidas *PlanetLab* como un conjunto representativo de la red de Internet total puede suponer problemas. La base de estas argumentaciones es que la mayoría de los nodos pertenecen a redes de investigación o educacionales [ZLPG05].

Además del problema descrito, existen otros dentro de la plataforma *PlanetLab,* como por ejemplo las deficiencias que en ocasiones surgen en el uso de sus herramientas de medidas. Estudios han demostrado que cuando una gran cantidad de usuarios están conectados a los nodos remotos a través de slices, se incrementa la carga y por lo tanto los retrasos aumentan. Estos retrasos pueden suponer una variación significativa a la hora de la toma de resultados de medidas realizadas a lo largo de la plataforma.

Se han desplegado a lo largo de la plataforma sistemas de monitorización distribuidos que proveen información de métricas de los recursos disponibles en los nodos como la memoria, uso de la CPU, etc. Un ejemplo de este rango de sistemas es Ganglia [MCC04].

Sin embargo, a pesar de los problemas existentes hay muchísimas razones por las que es positivo usar plataformas como la desplegada por PlanetLab, incluyendo disponibilidad, conjunto activo de nodos crecientes, aplicaciones, mecanismos de auto-políticas y herramientas.

En primer lugar hay que destacar los proyectos desplegados a lo largo de PlanetLab relacionados con medidas de Internet. Entre estos proyectos, es muy significativo el proyecto Scriptroute [SWA03], en el que los usuarios pueden conectarse de forma remota y hacer correr herramientas de medida, que son expresadas en forma de scripts. Entre todas las herramientas disponibles existe una gran variedad, desde herramientas de ancho de banda como bprobe hasta herramientas de medidas de RTT como ping.

En segundo lugar tenemos proyectos desplegados relacionados con las Content Distribution Networks. Los CDNs sirven como infraestructura de servidores de contenido distribuidos geográficamente. A lo largo de PlanetLab se implantó el sistema CoDeen [WPP+04] el cual permite ubicar contenido distribuido a lo largo de la infraestructura de la plataforma (admitiendo además el troceado para archivos grandes). Los nodos que participan en el sistema son monitorizados para eliminar a los de bajo rendimiento (alta tasa de pérdida de paquetes, bajo rendimiento DNS,…). Gracias a este sistema, no solo se consigue facilitar un nuevo servicio a los usuarios, sino que además proporciona la facilidad de detección de problemas internos.

Por otro lado PlanetLab también es usado para monitorizar fallos en los caminos de Internet. Por ejemplo PlanetSeer [ZZP+04], monitoriza las conexiones TCP a lo largo de CoDeen.

Por último destacar el uso de PlanetLab para medidas de estacionariedad en medidas de Internet. Muchas aplicaciones comparten componentes primitivos que pueden ser reutilizables por otras aplicaciones, por ejemplo, medidas de DNS. Se puede destacar el proyecto ATMEN [KMS05, KMV05], que consiste en la descarga masiva por parte de varios clientes de diferentes sitios Web populares durante el tiempo de dos semanas y media sin detenerse. Gracias a la herramienta httperf se puede rastrear el tiempo de búsqueda de DNS, tiempo de establecimiento de la conexión TCP y el tiempo de transferencia HTTP. Como uno de los resultados obtenidos de este estudio tenemos que la variabilidad del tiempo de establecimiento de conexión TCP y el tiempo de transferencia HTTP es nulo a lo largo de días, mientras que el tiempo de búsqueda DNS no. Gracias a proyectos como el citado, queda demostrada la aplicabilidad que tiene PlanetLab al estudio general de las diferentes propiedades de Internet.

2.4 Requisitos y restricciones

El proyecto descrito a lo largo de este documento se diseña e implementa en el contexto del proyecto europeo Leone. Este proyecto se encuentra actualmente en su fase inicial y sus funciones principales son la exploración de las soluciones actualmente existentes en los ámbitos de gestión de red, plataformas de medidas globales, visualización de medidas, medidas en Internet y calidad de experiencia.

Leone comienza en el despliegue de la infraestructura de pruebas en diferentes localizaciones de toda Europa a través de probes instaladas a lo largo de la red de Internet. Éstas tienen como objetivo arrojar luz sobre cuestiones de carácter relevante para los diferentes socios del proyecto.

Las probes están diseñadas para poder correr sobre ellas herramientas de medidas en Internet activas, aunque también se podrían realizar medidas pasivas si fuera necesario. Cada una de las probes lleva dentro herramientas por defecto como, por ejemplo, ping, aunque cada socio puede desarrollar una herramienta en forma de script, ubicarlo en la probe y lanzarlo para poder realizar las medidas que considere oportunas. Para esto se realizó el desarrollo inicial de un testbed para que los socios pudieran introducir herramientas desarrolladas por cada uno de ellos dentro de sus probes.

Esto tiene una serie de problemas en diferentes ámbitos: económicos, temporales, etc. Como hemos visto a lo largo del análisis del estado del arte, el desarrollo de muchas herramientas de medidas en Internet tienen la misma base o comparten ciertas medidas como, por ejemplo: medidas DNS. Para facilitar el trabajo se decidió realizar el diseño y desarrollo de un testbed en paralelo de carácter distribuido que completara las funcionalidades no existentes en el desarrollado inicialmente. Uno de los requisitos, por lo tanto, es que el testbed debe estar ligado a los socios del proyecto Leone, a partir de ahora usuarios, y a su infraestructura de medidas.

En el testbed desarrollado debe prevalecer el carácter distribuido. Para ello, se posibilita la oportunidad de subir las herramientas desarrolladas por cada usuario e incluso poder descargar y usar las herramientas de otros usuarios en tus probes o a lo largo de un sub-conjunto del total de probes de la infraestructura. Esto tiene un requisito asociado directamente: las probes pasan de ser entidades individuales a estar compartidas por todos los usuarios de la plataforma.

La posibilidad de realizar una planificación en la ejecución de las diferentes herramientas de medidas o tests se basa en la ejecución por defecto de cada probe. Por defecto, las probes realizan una serie de medidas cada un cierto tiempo – en concreto, cada dos horas, lanza una serie de herramientas básicas como http get, udp jitter, etc.-. Esto suscitó el interés para poder realizar una planificación de la ejecución de los tests desarrollados por cada usuario, no solo a nivel individual sino para poder, por ejemplo, medir el ancho de banda entre un sub-conjunto de probes de la infraestructura lanzando la misma herramienta desde cada una de ellas –tanto las propias de cada usuario como usando algunas de otros usuarios-. Esta planificación adicional hace necesario e indispensable algún tipo de calendario donde los usuarios puedan ver los diferentes tests programados, así como para que cada usuario pueda interactuar con él.

Para finalizar, con respecto al recuento de requisitos del proyecto, el desarrollo de esa planificación se basó en un requisito indispensable de cualquier plataforma o aplicación web: facilidad de uso y flexibilidad.

Con respecto a las restricciones de seguridad que debe superar el testbed desarrollado, se puede decir que no son necesarias de forma estricta, pues se ha desarrollado e implementado en el contexto del proyecto europeo Leone, que se realiza en un entorno de confianza colaborativo. Sin embargo, para evitar futuros problemas, se deben promover un mínimo control de uso.

Debido a la capacidad, por parte de los usuarios, para instalar tests utilizando la aplicación desarrollada en este proyecto se hace imprescindible controlar qué se instala y dónde. Un ejemplo relacionado directamente con este factor es el siguiente: un usuario desarrolla un test que lanza un tipo concreto de medida activa desde una de sus probes hacia ciertos destinos fijados por él para su análisis posterior. El análisis se basará en estudios teóricos que el propio usuario, posiblemente junto con su personal, ha desarrollado antes tomando los puntos de origen y destino de la prueba como puntos de partida. Si la aplicación no es capaz de controlar bien en qué probe se está instalando el test, se puede producir un desajuste entre lo esperado y la realidad de manera que haría el test inservible y haría del análisis, tanto previo como posterior, una pérdida de tiempo.

Otro factor a tener en cuenta en el control de uso es que los tests deben cumplir unos requisitos "*per-se*" como, por ejemplo, no pueden existir dos tests exactamente iguales dentro de la misma probe. Por ejemplo, si un usuario decide desarrollar un test similar al test HTTP_GET, el cual viene alojado por defecto en todas las probes, no solo desplegando su misma funcionalidad sino utilizando los mismos nombres de ficheros (ejecutable, script, etc.); la aplicación no debe permitir que se instale en las probes. Si no se realiza la comprobación pertinente puede suponer que un test desaparezca al ser suplantado por otro. Esto produciría graves consecuencias, pues si otro usuario decidiera posteriormente lanzar el test HTTP_GET que viene por defecto en las probes obtendría unos resultados totalmente distintos a lo esperado y volvería a suponer una pérdida de tiempo en los análisis, tanto previo como posterior.

La aplicación descrita a lo largo de este proyecto no está orientada a producto, sino que se trata de la implementación de una herramienta que facilite el desarrollo de del proyecto europeo Leone. Por lo tanto, deben realizarse pruebas en la aplicación de manera exhaustiva y precisa, así como verificar que cumple con las funciones descritas a lo largo de este documento. Si algún momento no se cumplen los requisitos descritos, podría perjudicar al desarrollo proyecto Leone.

2.5 Marco regulador

A lo largo de Internet se pueden encontrar una gran variedad de métricas definidas para la toma de medidas de la red así como herramientas que las lleven a cabo. El problema reside en el aumento de la dificultad a la hora de encontrar documentación que facilite información del modelo de la arquitectura o de los protocolos a cumplir al interconectar los diferentes dispositivos cuando se está realizando tomas de medidas con miles de nodos desplegados a lo largo de la red.

Se están realizando grandes esfuerzos por parte de la comunidad de investigación por definir estándares a la hora de realizar plataformas de medidas a gran escala (LMAP). Esto se puede ver reflejado, por ejemplo, en los diferentes drafts publicados por la organización [IETF] donde intentan arrojar luz sobre diferentes cuestiones y en concreto sobre las LMAP. Los Internet-Drafts [IETFaid] son documentos de trabajo validados por IETF que son válidos durante seis meses y, una vez transcurrido dicho período, los documentos deben ser actualizados, reemplazados o declarados como obsoletos.

Dentro de los drafts publicados por la organización IETF se puede encontrar diversos documentos en relación a las LMAP, por ejemplo, [EBM13]. A lo largo del documento se presenta un marco global para las medidas a gran escala y discute qué elementos pueden ser estandarizados por la organización IETF. Otro ejemplo es [EMB+13], donde se define la terminología para las LMAP.

El diseño del proyecto descrito a lo largo de este documento se basa en el draft [SJM12] donde se describe la arquitectura lógica para realizar las medidas a gran escala y se resumen los requisitos más importantes para los protocolos que se debe seguir a la hora de interconectar los diferentes dispositivos de la arquitectura. El documento está sujeto al [Bra05a] y al IETF Trust´s Legal Provisions Relating to IETF Documents [TLP] y está validado en plena conformidad con [Bra05b].

Este documento ha sido elegido como referencia ya que, a diferencia de otros documentos de trabajo donde hacen un análisis simple de las LMAP, éste presenta un estudio en profundidad no solo a la hora de definir los distintos elementos, sino yendo más allá y especificando una lista de requerimientos y casos de uso para el correcto uso de una LMAP.

A lo largo del draft se propone una arquitectura de carácter global haciendo especial hincapié en los requisitos funcionales y de seguridad para los protocolos encargados de conectar los elementos de la arquitectura, los cuales harán posible construir capacidades de medidas en redes hogares, routers de borde, etc. Las propuestas de diseño e implementación son válidas tanto para redes cableadas como inalámbricas. Sin embargo deja de lado un factor también importante como la autenticación en el acceso.

Para la toma de medidas activas el cliente necesita un servidor para que actúe como receptor del tráfico de medidas. Dado que se van a realizar medidas a gran escala, se asume que un controlador de medidas provee con la información necesaria a los clientes sobre qué medir o cuándo realizar las medidas. En algunos casos se necesita también una entidad que actúe como recolector de los datos resultantes de la toma de medidas para su posterior análisis. Por ejemplo, la toma de medidas del rendimiento de TCP depende del algoritmo de congestión de TCP. El controlador, en este caso, enviaría los parámetros al cliente y el recolector de datos debería ser capaz de determinar sin ambigüedad qué parámetros fueron usados para ese conjunto específico de muestras. A continuación se profundizará un poco más en cada uno de los diferentes elementos descritos.

El nodo cliente es el punto de referencia para las medidas enviando tráfico de medidas a otros elementos (medidas activas) o bien observando el comportamiento de ciertas métricas en la red (medidas pasivas). Su funcionalidad debe estar implementada en función de los diferentes contextos de usuario y debe proveer comunicación entre los diferentes segmentos de la red (por ejemplo entre los abonados de una red y la red ISP).

El nodo servidor solo es necesario para medidas activas que requieran de un segundo nodo de medidas que genere tráfico o actúe como sumidero. Los clientes deben usar los diferentes servidores posibles eligiendo, por ejemplo, al servidor más cercano.

El nodo controlador provee información necesaria a los nodos clientes para la realización de las medidas, por ejemplo; qué medidas se debe realizar, contra qué nodos servidores, cuándo realizarlas o qué datos serán recopilados por el nodo recolector. También puede servir para indicar a los nodos clientes una planificación, automatizando así la toma de medidas de la red. Además de estas funcionalidades el nodo controlador debe ser capaz de interactuar con otros nodos controladores.

El nodo recolector de datos recoge los resultados de la toma de medidas por parte de los nodos clientes, los cuales pueden ser almacenados de diferentes maneras, por ejemplo, como una base de datos o almacenamiento FTP. En función del proyecto en el que se encuentre los datos serán accesibles para toda la comunidad de Internet o solo para usuarios autorizados (en cuyo caso será necesario implementar métodos de autenticación para el acceso a los datos recopilados).

En algunos casos puede ser útil tener un servidor de parámetros de la red que informe a las diferentes entidades sobre qué tipo de tecnología se están utilizando para la toma de medidas, por ejemplo, FTTP; en otros casos también puede ser de utilidad conocer la velocidad nominal teórica que la red es capaz de ofrecer.

Los diferentes elementos descritos deben interactuar entre ellos y para hacerlo de una manera lógica se han definido una serie de protocolos a seguir en dicha comunicación.

Las métricas usadas en las medidas activas entre dos nodos (cliente-servidor) son particulares de cada medida; en concreto, de qué se esté midiendo. Por ejemplo, una métrica de VoIP deberá usar un conjunto definido de paquetes UDP para estimar su rendimiento.

Periódicamente los nodos clientes deberán solicitar a los nodos controladores actualizaciones de la planificación de la toma de medidas: tipos de medidas que el nodo cliente puede realizar, contra qué servidores lanzar las medidas y qué planificación seguir para la realización de dichas medidas. El intervalo de tiempo entre estas actualizaciones puede variar en función de las necesidades, por ejemplo, cada dos horas, cada día o, incluso, cada semana.

Un nodo controlador puede requerir a otro nodo controlador que realice un programa específico de medidas indicando tests específicos, planificaciones y parámetros necesarios. Además de ésta puede intercambiarse otro tipo de información como, por ejemplo, su identidad para la verificación de la solicitud, un identificador específico de test, etc.

Entre las diferentes tomas de medidas por parte de los nodos clientes, éstos enviarán los resultados de dichas muestras a las entidades recolectoras de datos. Las muestras enviadas deberán ir clasificadas con información como cuándo fueron recolectadas, información del dispositivo frontera y qué host de medida fue usado. En el caso de tomas de medidas móviles podría ser útil incluir información acerca de la localización de la toma de medidas (lo cual habría que hacer con precaución pues puede invadir la privacidad del usuario).

En algunas ocasiones puede ser de interés que el nodo cliente interactúe con el servidor de parámetros de red para obtener información acerca de valores nominales de los parámetros de los diferentes servicios, dirección MAC o identificadores de los clientes móviles como sus IMEI. Los datos que debe devolver dicho servidor deben incluir información como las velocidades nominales de subida y bajada, cuotas de datos y tecnología de las capas de datos y física.

La colaboración entre el servidor de medidas, el nodo controlador y el nodo encargado de la recolecta de datos es imprescindible para la correcta configuración de los parámetros por diferentes razones como, por ejemplo, disponibilidad o seguridad.

La toma de las medidas debe iniciarse en los nodos clientes o, en su defecto, por parte de los nodos controladores. El hecho de que los nodos clientes comiencen la comunicación puede evitar diversos problemas (por ejemplo los relacionados con NATs o cortafuegos). Sin embargo, también se contempla el caso en el que el proveedor desee iniciar alguna medida puntual o cambiar los parámetros de medida antes de que el cliente contacte con el controlador. Para estos casos puede ser útil usar un mecanismo publicación-suscripción.

Además de los protocolos descritos se han clasificado unos requisitos específicos de cada una de las entidades que a continuación se detallarán comenzando por los requisitos relacionados con la plataforma en general:

- La arquitectura debe permitir medidas instantáneas iniciadas por usuarios finales, medidas muestreadas iniciadas por los proveedores de red y medidas por uno o más terceros.
- Los nodos clientes y servidores deben soportar un conjunto extensible de métricas de rendimiento.
- Los nodos clientes, nodos servidores y recolectores de datos deben ser operados por diferentes entidades administrativas, incluyendo otras que no sean el proveedor de servicios de Internet.
- Los nodos clientes deben ser capaces de realizar medidas activas y pasivas.
- Todas las entidades deben ser capaces de autenticar las entidades con las que se comunican.
- Las muestras de las medidas deben ser asociadas inequívocamente con los parámetros de medidas, así como por referencia o por su valor.
- Para asegurar disponibilidad y escalabilidad, las implementaciones deben ser capaces de implementar múltiples controladores, nodos servidores y recolectores de datos con su apropiado balanceo de carga y migración tras error.

En segundo lugar se enumeran los requisitos que afectan directamente a los nodos clientes:

- La arquitectura debe permitir la participación de un nodo cliente en particular en una o más plataformas de medidas independientes.
- Un nodo cliente debería ser capaz de cambiar automáticamente a otro servidor de medidas si uno no responde.
- Un nodo cliente debe ser capaz de registrarse con la colección de datos de la plataforma automáticamente, anunciando su disponibilidad y los parámetros de sistema relevantes.
- Un nodo cliente debe ser capaz de anunciar qué tipo de medidas debería poder realizar, por ejemplo, mediante identificadores de medidas.

Por otro lado se han definido los requisitos de los nodos controladores:

- El sistema de medidas debe soportar medidas que son planificadas previamente por un calendario.
- El nodo controlador debe ser capaz de especificar el intervalo en el que desea ser contactado para actualizar las planificaciones de las medidas.
- Un nodo cliente debería ser capaz de descubrir automáticamente controladores provistos por su proveedor de servicios de Internet.
- Un nodo cliente debe ser capaz de autenticar y autorizar al nodo controlador.
- Los datos intercambiados entre el cliente y el controlador deben permitir ser encriptados opcionalmente y usar mecanismos para proteger su integridad.

Los nodos encargados de la recolección de datos deben cumplir los siguientes requisitos:

- El protocolo de mensajes para muestras de medidas debe permitir nuevos tipos de medidas y parámetros.
- El nodo debe poder proteger su integridad y confidencialidad de los datos de medidas intercambiados entre el cliente y el nodo recolector.
- El protocolo de datos intercambiados entre servidores y nodos recolectores deberían poder permitir la definición de elementos de datos comunes.
- El cliente de medidas debería ser capaz de conmutar automáticamente por error para alternar recolectores de datos.
- Los clientes deben ser capaces tanto de enviar datos inmediatamente o de retrasar el envío de los datos al nodo recolector.
- Los clientes deben ser capaces de entrelazar muestras de datos de diferentes métricas de medidas al nodo recolector.
- El nodo recolector debería ser capaz de determinar si el reloj del cliente está sincronizado con el suyo.
- Los datos intercambiados entre el cliente y el recolector deben estar sujetos a controles de flujo y congestión.
- Los clientes deben ser capaces de determinar que están iniciando sesión con el nodo recolector deseado en vez de un impostor.

Por último se debe cumplir los siguientes requisitos en las funciones necesarias para obtener los parámetros de la red:

- Los clientes deberían ser capaces de obtener valores nominales de parámetros de servicios de la red en formatos legibles, así como anunciar velocidad y latencia típicas.
- El conjunto de parámetros de la red debe ser extensibles y compatibles con versiones anteriores.

- El nodo cliente debería ser capaz de determinar los parámetros de la red del nodo servidor sin configuración manual.
- El protocolo entre el nodo cliente y el nodo encargado de los parámetros de la red debería poder soportar una variedad de identificadores de clientes, así como de direcciones de red, direcciones de la capa de enlace, identificadores AAA o identificadores hardware.
- Los datos intercambiados entre el servidor de parámetros de la red y el cliente deberían poder asegurar su confidencialidad e integridad.
- El protocolo debería soportar funcionalidad de autenticación para restringir el acceso a los parámetros de la red solo a nodos autorizados.
- La entidad que solicite información al servidor de parámetros de la red debe ser capaz de asegurarse a sí misma que se está comunicando con un servidor auténtico.
- Los clientes del servidor de parámetros de red deberían ser capaces de ser informados automáticamente si hay cambios en los parámetros.

La seguridad a lo largo de la infraestructura y durante la interacción entre sus diferentes elementos es crucial tanto para rechazar el acceso de terceros que quieran hacer un uso malicioso de alguna de las entidades o de la infraestructura en sí; como de asegurar la encriptación y seguridad en las transmisiones de datos con información sensible de los usuarios.

Capítulo 3

Diseño y desarrollo de la solución

3.1 Introducción

A lo largo de este capítulo se explica, con el máximo nivel de detalle posible, el problema que se desea resolver, explicando el marco en el que se desarrolla y justificando las diferentes decisiones de diseño. Además, se detallan las tecnologías utilizadas para llevar a cabo el desarrollo de la solución, intentando justificar cada componente por separado.

En el primer apartado se realiza una introducción intentando aclarar qué es lo que se desea desarrollar en este proyecto, recopilando los puntos más importantes que enmarcan el mismo. A continuación, se pasa a realizar un estudio exhaustivo de los diferentes elementos que componen la solución desarrollada, enmarcados por la infraestructura de medidas a gran escala del socio Samknows. Para finalizar la primera parte, se recogen los diferentes casos de uso derivados del diseño de la solución, tanto los casos finalmente implementados como los que han quedado para un futuro trabajo.

En el segundo apartado, se realiza un estudio de los diferentes componentes que, finalmente, han sido elegidos para el desarrollo del testbed diseñado en la primera parte. Se comienza por explicar, de forma específica, cómo funciona la plataforma en la que se

enmarca este proyecto (la plataforma de medidas de Samknows), para luego pasar a los módulos específicos desarrollados: lenguaje de programación, servidor, base de datos, arquitectura, librerías de interés y herramientas de desarrollo. Además, se ha descrito un pequeño estado del arte de cada componente para justificar cada una de las partes del proyecto.

3.2 Diseño

El objetivo principal del proyecto es el diseño de un testbed enmarcado dentro del proyecto europeo Leone, con el objetivo principal de dar soporte en la fase inicial del mismo. Un testbed se define como una plataforma para la experimentación de desarrollo de grandes proyectos. Por lo tanto, un testbed debe permitir el testeo riguroso, transparente y reproducible de teorías científicas, herramientas computacionales y nuevas tecnologías. En concreto, los testbed son usados para la realización de pruebas en entornos confiables o de producción, permitiendo realizar un testeo aislado del entorno en el que realmente se va a llevar a cabo el proyecto final. Un esqueleto es implementado para que funcione como parte del sistema completo, pero funcionando como un entorno limitado. Un ejemplo son las páginas web que permiten realizar pruebas con CSS o HTML sobre ellas para obtener el resultado final en un entorno controlado.

El proyecto Leone consiste, en general, en el despliegue de una gran cantidad de probes a lo largo de Europa para realizar medidas activas a gran escala y obtener resultados beneficiosos para todos los socios participantes. Un resumen de las ideas principales del proyecto Leone es:

- Nuevos tests corriendo en las probes, para medir la calidad de experiencia (QoE). Se basan en la plataforma global del socio Samknows, el cual cuenta con miles de probes en los hogares de clientes de banda ancha, desplegando 100 probes en las redes de los socios ISPs.
- Análisis para determinar la causa de la degradación de la calidad de experiencia (QoE). La herramienta combina múltiples puntos de vista provistos por los datos multi-dimensionales, con el fin de aislar qué parte de la infraestructura de servicios de Internet ha fallado o necesita ser actualizada.
- Visualización de los resultados de las medidas y análisis.
- Integración de las medidas y de los resultados de los análisis en herramientas de administración de redes, ejemplificados por la herramienta del socio MG-Soft.
- Reparación automática del problema, donde sea posible en algunos escenarios.

Tras el análisis del proyecto Leone (marco en el que se desarrolla, estructura global, etc.) se decide implementar un testbed cuyo objetivo es realizar pruebas con tests a menor escala antes de pasarlos a producción en la plataforma de medidas implementadas por el socio Samknows. Esto permite una serie de grandes ventajas: independencia de la plataforma de Samknows en el desarrollo de los tests, facilitar la colaboración entre los investigadores durante la fase de desarrollo, etc.

Para la implementación del testbed sobre un entorno real surgen una multitud de posibilidades. Así pues, tras realizar un exhaustivo estudio del estado del arte se opta por realizar el diseño basándose en el draft [SJM12]. De esta manera, tanto los componentes del testbed, como los protocolos para la interacción entre ellos, toman el documento como un modelo de referencia.

3.2.1 Elementos del testbed

En el pasado capítulo anterior, se ha realizado un completo estudio y desglose de los contenidos del draft donde, entre otros aspectos, se definen los elementos necesarios para desarrollar una plataforma de medidas: nodo cliente, nodo servidor, nodo controlador, nodo recolector y, de manera opcional, un nodo encargado de los parámetros de la red.

A continuación se detallan los elementos que componen el diseño definitivo:

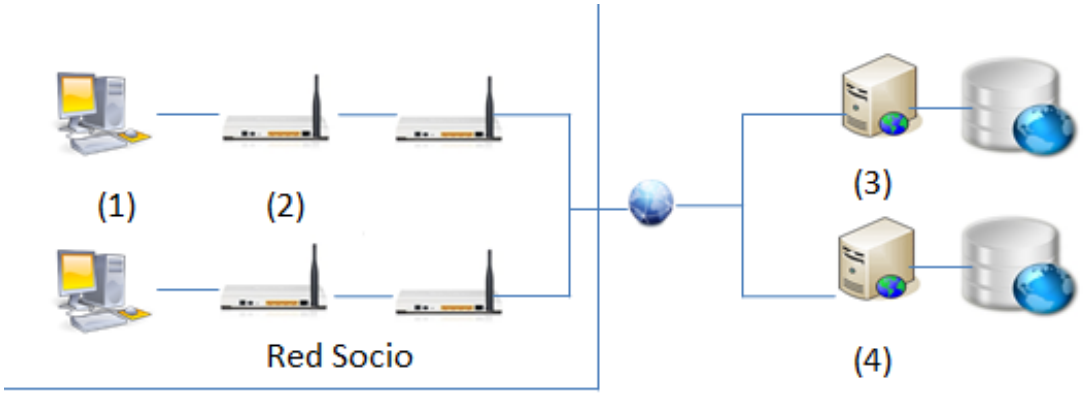

Ilustración 1. Infraestructura del testbed

- **Usuarios finales (1):** se entienden por usuarios finales como los que alojan las probes en sus redes hogares para las pruebas a gran escala a lo largo de la segunda fase del proyecto Leone.
- **Probes (2):** son repartidas entre los socios del proyecto Leone formando una gran red de medidas en Europa. Actúa tanto de nodo cliente como de nodo servidor en función de la necesidad del test en el momento oportuno.
- **Servidor central (3):** nodo encargado de controlar las actualizaciones de todas las probes. La comunicación siempre es iniciada por los nodos clientes por lo que podemos obviar posibles problemas como los surgidos por el uso de firewalls o NATs.
- **Aplicación web (4):** interfaz amigable para que los socios compartan tests, suban tests a sus probes, eliminen tests de las probes o incluso planifiquen su lanzamiento.

- **Socios:** diferentes entidades que, gracias a su colaboración, hacen posible este proyecto de medidas a gran escala. UC3M, BT, Samknows,…. En el caso de proveedores, como BT, son las encargadas de repartir las probes entre sus clientes finales para la segunda fase del proyecto Leone.

3.2.2 Casos de uso

Una vez explicados los diferentes elementos de los que se compone el testbed desarrollado, a continuación se detallan los diferentes casos de uso:

- **Caso de uso 1**: Instalación de nuevos tests
- **Caso de uso 2**: Borrado de un test existente
- **Caso de uso 3**: Planificar tests para que sean lanzados
- **Caso de uso 4**: Lanzar un test desde la web
- **Caso de uso 5**: Cross Probing
- **Caso de uso 6**: Registro de usuario

3.2.2.1 Caso de uso 1: Instalación de nuevos tests

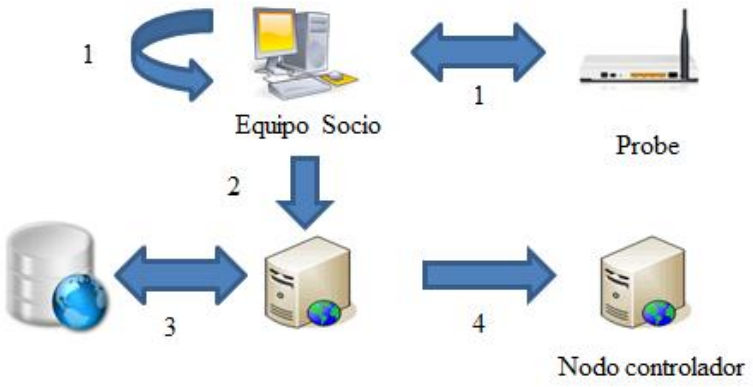

Ilustración 2. Diseño del caso de uso 1

En este caso de uso, se plantea el procedimiento a seguir en el caso de que un socio registrado quiera compartir un test con el resto de socios. En primer lugar el socio debe realizar el diseño y desarrollo del test; éste se compone de uno o varios archivos binarios y de uno o varios scripts encargados de la ejecución de los binarios. Para este proceso cada socio debe seguir las indicaciones de la wiki, donde se muestra cómo proceder para desarrollar un nuevo test. Para el siguiente paso se supone que el socio, además de desarrollar el nuevo test, ha sido capaz de probar su funcionalidad tal y como se indica en la wiki.

En segundo lugar el socio deberá autenticarse en la web, seleccionar el apartado correspondiente a subir un nuevo test y rellenar un cuestionario con los datos necesarios para que la subida se haga efectiva: nombre del test y descripción de funcionalidad.

La infraestructura web desarrollada se encarga de almacenar los datos relevantes en la base de datos (nombre del test, fecha, descripción, usuario). Tan sólo los datos relevantes de los tests son almacenados en una base de datos para llevar el control del almacenaje.

Una vez que los datos son almacenados en la base de datos, se procede a la conexión entre el nodo controlador de Leone y la infraestructura web para copiar los ficheros correspondientes al nuevo test de manera que éstos sean descargados en la próxima actualización por todas las probes.

Inicialmente, se piensa en un modelo de instalación selectiva donde cada socio pudiese instalar solo los tests desarrollados por sí mismos o mediante invitación a otros. Finalmente, por simplicidad, se llega a la conclusión de que lo mejor es que, una vez se haya subido un test, éste sea compartido por todos los participantes del proyecto alojando los archivos pertinentes en todas las probes de la infraestructura.

Otra opción que se tiene en cuenta inicialmente es la posibilidad de añadir visibilidad al alojamiento de tests en cada probe. De esta manera, algunos tests serían visibles para todos y otros solo para aquellos que lo suban y no quieran compartir esa información con el resto. Esto también se desechó debido diferentes factores: aumento de la complejidad y, principalmente, por la falta de necesidad al encontrarse el proyecto en un entorno seguro de colaboración.

En cuanto a las posibilidades de descarga de los archivos correspondientes al nuevo test, también se barajea inicialmente realizar la descarga en las probes solo de los archivos pertinentes al nuevo test o sustituir toda la carpeta "/tmp" en cada actualización. Puesto que la empresa Samknows ya había realizado una implementación en su nodo controlador para la descarga de archivos basado en una estructura jerárquica de ficheros y en actualizaciones periódicas, se desecha la idea de realizar la descarga de todo el contenido de la carpeta "/tmp".

En un principio también se piensa en la posibilidad de añadir una funcionalidad al servidor web para que se conecte directamente con cada probe y le envíe los archivos pertinentes. Finalmente, se decide hacer uso del nodo controlador implementado por Samknows.

También cabe destacar otra idea desechada, tener un control de versiones de los tests subidos por los socios. Finalmente se decide por simplificación, que una nueva versión de un test existente, debe ser cargada como un nuevo test.

Para acabar de dar una vuelta por las ideas desechadas acerca de este primer caso de uso, se barajea la posibilidad de aplicar algún tipo de código hash a los tests para eliminar suplantaciones de identidad y que dos socios quisieran subir el mismo test con otro nombre. Esta idea fue desechada por simplicidad aunque podría ser la base de una futura extensión.

3.2.2.2 Caso de uso 2: Borrado de un test existente.

A continuación, se describen los pasos a seguir para realizar el borrado de un test existente de todas las probes:

1. El usuario (socio) se conecta a la web y realiza el proceso de autenticación. Después, selecciona la opción de borrado de un test existente subido previamente por él.
2. La infraestructura web recoge la petición y borra los datos pertinentes de la base de datos.

3. La infraestructura web se conecta al nodo controlador para realizar el borrado de los archivos pertinentes a dicho test. En la siguiente actualización se borrarán dichos archivos de todas las probes.

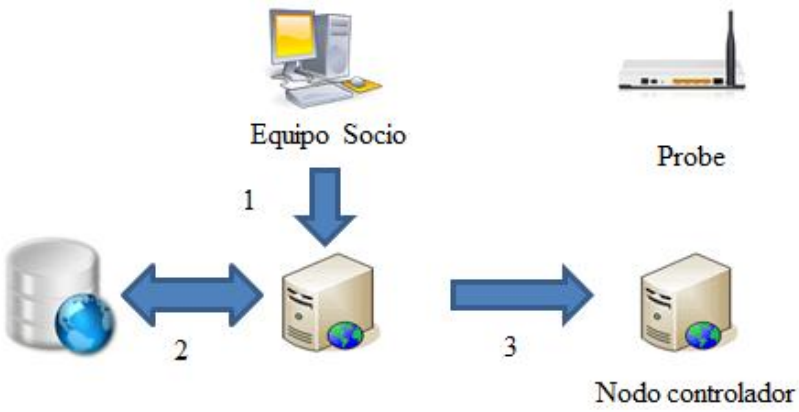

Ilustración 3. Diseño del caso de uso 2

En un principio se plantea que todo usuario pueda borrar cualquier test subido a la plataforma pero finalmente por seguridad solo se permite realizar la acción de borrado a aquellos usuarios que hayan subido el test.

Otra idea desechada es la opción del borrado selectivo. Esta opción consiste en que, una vez haya seleccionado un usuario la opción de borrado de alguno de sus tests, dar la opción al resto de borrar el test de sus probes o mantenerlo guardado dentro del sistema de ficheros propios de su probe.

El diseño seguido para este caso de uso tiene un inconveniente: dada la posibilidad de planificar tests, si un test tuviese planificada su ejecución, se va a obtener como resultado un error al no encontrar los archivos solicitados. Para solucionar este inconveniente, existen dos opciones: el usuario puede mirar dónde lo ha planificado y realizar los cambios que considere necesarios, o bien realizar un sistema de análisis sintáctico de ficheros que compruebe el contenido de cada planificador para eliminar cualquier rastro del test.

3.2.2.3 Casos de uso 3: Planificar tests para que sean lanzados

La idea principal de este caso de estudio, es que las probes tienen por defecto un lanzamiento automatizado de una serie de tests cada 2 horas. Esto lleva a pensar en la posibilidad de migrar esa planificación a cualquier hora del día y a cualquier test alojado en la probe. El diseñó quedaría:

1. El usuario se conecta a la web y realiza el proceso de autenticación.
2. El usuario accede al calendario web donde ve las planificaciones de los tests y escoge un tiempo de inicio del test que quiera correr en un slot vacío del calendario. Al escogerlo, saldrá una notificación para que el usuario elija primero en qué probes quiere lanzar el test y en segundo lugar qué test quiere lanzar.

3. Una vez el usuario haya seleccionado todo, la infraestructura web se conectará al nodo controlador y modificará los archivos pertinentes para que esa ejecución sea llevada a cabo.

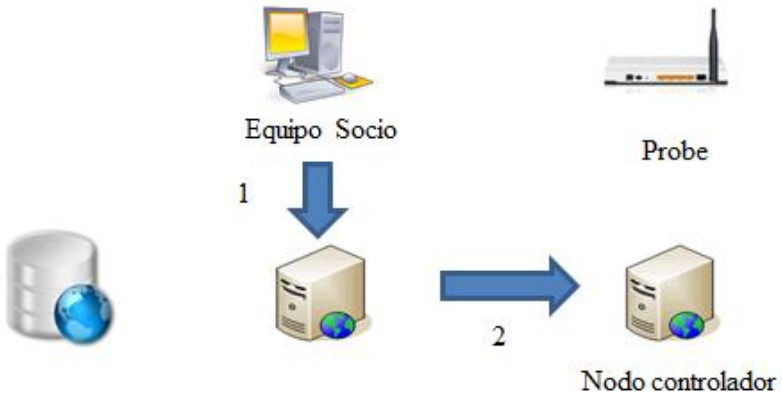

Ilustración 4. Diseño del caso de uso 3

Para la realización de esta planificación se han planteado gran variedad de posibilidades. La planificación se basa en ciertos archivos alojados en las probes ("*jobfile*", "*scheduler*", etc.) los cuales llevan intrínsecos en sus líneas qué tests deben llamar. Una posibilidad es almacenar las diferentes versiones de planificadores que el usuario desee para que el éste pueda escoger cual quiere instalar en ciertos momentos y modificarlos a su antojo. Otra posibilidad es ir editando el contenido de los archivos (añadiendo y eliminando líneas) en función de los tests que se deseen lanzar. La idea más sencilla es que haya unos archivos por defecto en todas las probes y en función del momento concreto las líneas se vayan añadiendo o eliminando.

3.2.2.4 Caso de uso 4: Lanzar un test desde la web

Este caso de uso se basa en la posibilidad de que un usuario desee lanzar un test de forma instantánea y remota desde la plataforma web desarrollada.

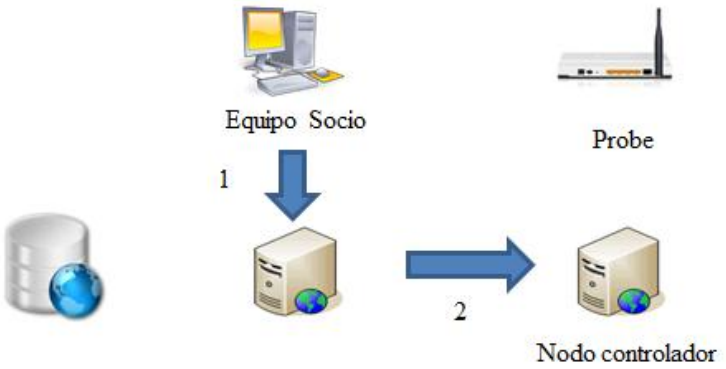

Ilustración 5. Diseño del caso de uso 4

En este caso, lo primero que debería realizar el usuario es la conexión a la web y el proceso de autenticación. Una vez lo haya superado, escogería el test que deseara lanzar y seleccionaría la opción correspondiente a lanzar el test. Puesto que los usuarios pueden tener varias probes, deberá seleccionar también en cuál de ellas desea lanzar el test.

Una vez realizado este proceso, la infraestructura web se conectaría con el nodo cliente que el usuario hubiera escogido y lanzaría el test ofreciendo los resultados obtenidos.

Este caso de prueba no se ha implementado aunque puede ser implementado en el futuro extendiendo la funcionalidad de la plataforma.

3.2.2.5 Caso de uso 5: Cross Probing

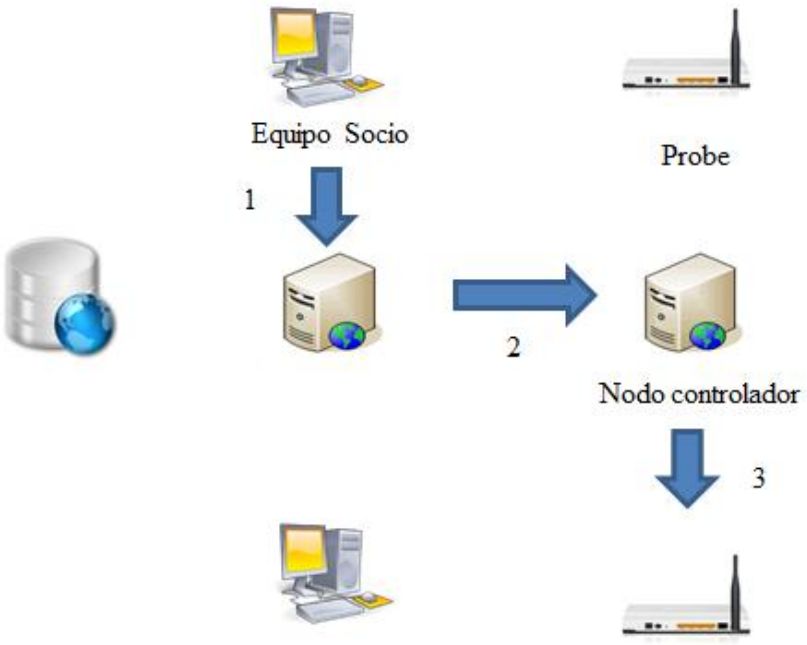

Ilustración 6. Diseño del caso de uso 5

Este caso fue inicialmente propuesto como una posibilidad de realizar testing distribuido.

Inicialmente, el usuario debe acceder a la web y realizar el proceso de autenticación. Una vez realizado el primero paso, debe seleccionar el test que desea lanzar y en qué probe desea lanzarlo. El usuario propietario de la probe sería notificado de esta "invitación" y a partir de ese momento podrían ocurrir tres cosas:

1. Que el usuario no responda y, por lo tanto, no pase nada.
2. Que el usuario destino lo rechace, siendo notificado el usuario origen.
3. Que el usuario destino acepte. En este caso el test sería lanzado y monitorizado de ahí en adelante por parte del usuario origen.

Este caso fue propuesto y desechado, con una gran cantidad de posibilidades de implementación y variaciones, en el congreso de Leone de Roma en Febrero. Las razones de su rechazo fueron, entre otras, la falta de privacidad en su desarrollo.

3.2.2.6 Caso de uso 6: Registro de usuario

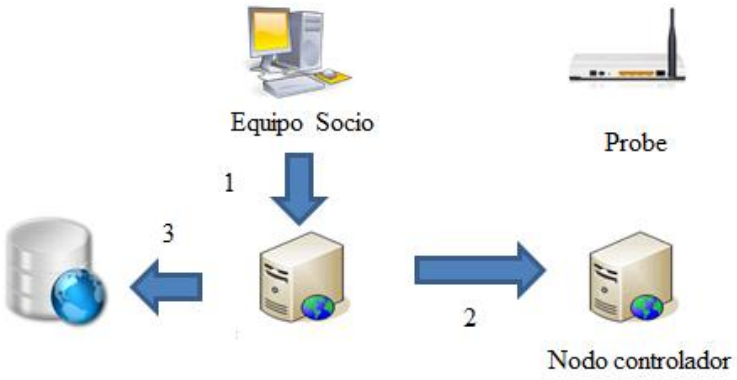

Ilustración 7. Diseño del caso de uso 6

Este último caso de uso describe el proceso de registro en la aplicación web. Una vez diseñado y expuesto en el congreso de Leone de Roma en Febrero, fue rechazado por los diferentes participantes del proyecto por varias razones, destacando la falta de utilidad puesto que nos encontramos en un entorno colaborativo y cerrado; es decir, no se prevé que, a corto plazo, se incorporen nuevos participantes al proyecto.

En primer lugar el usuario se conecta a la web. Acto seguido debe seleccionar la opción de registro y rellenar un pequeño formulario con su nombre y la(s) IP pública(s) de su(s) probe(s).

A continuación, el administrador obtiene la petición de registro y se encarga de generar una contraseña así como de validar al nuevo usuario de la aplicación. El administrador también debe actualizar la base de datos con los datos del usuario y su contraseña.

Una posibilidad de diseño es eliminar el papel del administrador y utilizar un validador web (mandando al usuario una URL para la confirmación de registro). En este caso, por razones de seguridad, se recomienda el uso de la figura del administrador.

3.3 Desarrollo

A continuación, se detallarán las diferentes etapas y componentes del desarrollo llevado a cabo del proyecto descrito en este documento, dando una visión más específica de cómo ha sido llevado a cabo el trabajo de desarrollo en sus diferentes ámbitos: lenguaje de programación, entorno, librerías utilizadas, etc. Además, se lleva a cabo un análisis exhaustivo de cómo realizar los casos de uso descritos en el capítulo de diseño, intentando arrojar luz sobre las dudas que puedan haber surgido.

3.3.1 Java

Java es una tecnología usada para el desarrollo de aplicaciones que convierten a la Web en un elemento más interesante y útil [JAVA]. Java es la base para multitud de aplicaciones en red y es el estándar mundial para el desarrollo y distribución de software empresarial, juegos y aplicación móviles, además cuenta con un fuerte apoyo de herramientas [JAVAb].

Para ejecutar cualquier aplicación que requiera Java basta con descargar el fichero ejecutable de la página oficial [JAVAc] y proceder a su instalación.

Para poder desarrollar, Java tiene un completo paquete descargable en su página web. El JDK es un entorno de desarrollo para construir aplicaciones, applets y componentes utilizando el lenguaje de programación Java, éste incluye herramientas útiles para desarrollar y probar programas escritos en el lenguaje Java y se ejecuta en la plataforma Java [JAVAd].

Debido a su gran utilidad y su uso tan extendido en la programación de aplicaciones web, se decide usar para la implementación de la arquitectura web descrita en este documento. No solo para su funcionamiento sino, también, a lo largo del desarrollo e implementación del mismo.

3.3.2 Servidor: Apache Tomcat

Apache Tomcat es una implementación software de código abierto de las tecnologías Java Servlet y JavaServer Pages (JSPs). Las especificaciones de Java Servlet y JavaServer Pages son desarrollados bajo el Java Community Process [TOM].

Apache Tomcat se desarrolla en un entorno abierto y colaborativo publicado bajo la Apache License versión 2. Apache Tomcat tiene la intención de ser una colaboración de los mejores desarrolladores en todo el mundo y cualquiera puede participar [TOMb].

La mayoría de los frameworks web Java actuales pueden ser lanzados sobre Apache Tomcar, por ejemplo: Spring [SPR] o Struts [STR]. Además, Apache Tomcat puede ser usado como servidor HTTP aunque se recomienda el uso del servidor Apache HTTP.

Para instalarlo en cualquier máquina Windows, se necesita el instalador de [TOMc]. Para una distribución Linux basta con lanzar por línea de comandos las siguientes líneas de código:

1. *Apt-get install tomcat7*
2. *Apt-get install tomcat7-admin*

Una vez descargado e instalado, se deben configurar las variables de entorno:

- *CATALINA_HOME*: debe apuntar al directorio donde esté ubicado el servidor Tomcat.
- *JAVA_HOME*: debe apuntar al directorio donde esté instalado Java.
- *PATH*: debe apuntar a "*${JAVA_HOME}/bin*".

En Windows, se debe ejecutar el archivo "*tomcat7.exe*" en la carpeta "*bin*" para que el servidor Tomcat sea lanzado. En Linux, por el contrario, se arranca con el script "*startup.sh*" en "*${CATALINA_HOME}/bin"* y pararlo con el script "*shutdown.sh*".

Una manera de probar que se ha lanzado correctamente es intentar acceder a la URL http://localhost:8080. Si es posible el acceso, es que está funcionando correctamente. Si el servidor no arranca, una de las causas más probables es que el puerto en el que está escuchando el servidor Tomcat esté ocupado por otra aplicación. El puerto de escucha es fácilmente editable en el fichero "*${CATALINA_HOME}/conf/server.xml*".

Para la configuración de los diferentes parámetros Tomcat proporciona una consola de administración basada en web con una interfaz amigable: http://localhost:8080/manager/html. Para acceder, en Windows basta con indicar el usuario y la contraseña; en Linux, ha de añadirse manualmente el usuario al fichero "*tomcat-users.xml*".

Dentro del sistema de directorios se debe destacar:

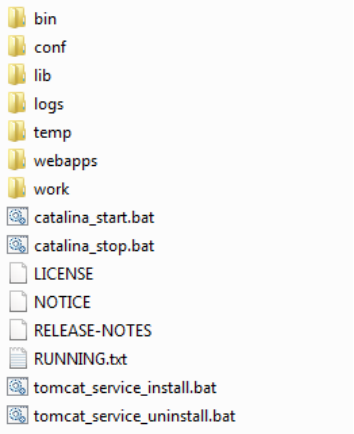

Ilustración 8. Sistema de directorios de Apache Tomcat

- ***/bin*:** startup, shutdown, y otros scripts. Los archivos con extensión "*.sh*" (para sistemas Unix) son duplicados funcionales de los ficheros "*.bat*" (para sistemas Windows)
- ***/conf*:** los archivos de configuración y DTDs relacionados. Destaca el archivo "*server.xml*", el cual es el fichero de configuración principal para el contenedor.
- ***/logs*:** archivos de log.
- ***/webapps*:** donde se alojan las aplicaciones web desarrolladas por terceros.

La decisión de utilizar el servidor Apache Tomcat para el alojamiento de la infraestructura web del proyecto se basa en su gran rendimiento, facilidad de uso. Además al tratarse de un servidor globalmente extendido para el desarrollo web facilita la exportación a otros entornos y reduce las barreras de entrada para un futuro trabajo con otros desarrolladores.

3.3.3 Base de Datos: MySQL

En primer lugar se hará una introducción de las bases de datos relacionales. Éstas son colecciones de datos almacenados en una o más tablas que, a su vez, constan de filas y columnas y pueden estar relacionadas entre sí. Para gestionarlas se usan Sistemas Gestores de Bases de Datos Relacionales entre las que destacan Oracle Database, Microsoft SQL Server y MySQL.

El lenguaje SQL es un lenguaje estándar para usar y gestionar bases de datos relacionales usado en los diferentes sistemas de gestión [R10].

MySQL es un sistema gestor de bases de datos relacional, multiusuario y multihilo más popular en todo el mundo que, además, es software libre. Se ofrece bajo la licencia GNU GPL para su uso personal, para empresas se debe pagar una cantidad. Muchas de las organizaciones más grandes y de más crecimiento del mundo, incluyendo Facebook o Google, se basan en MySQL para ahorrar tiempo y dinero [MyS].

La gestión de la base de datos MySQL en este proyecto se ha apoyado en la herramienta phpMyAdmin [PHPMA]. Se trata de una herramienta gratuita desarrollada en el lenguaje de programación PHP con interfaz web que facilita la administración de bases de datos MySQL. PhpMyAdmin es compatible con un gran rango de operaciones en MySQL: gestión de diferentes bases de datos, tablas, campos, relaciones, usuarios, permisos, índices, etc. PhpMyAdmin además aporta una gran variedad de documentación [PHPMAb] y es posible participar en su proyecto a través de sus páginas wiki [PHPMAc].

A continuación se detalla el diagrama de entidad / relación de la base de datos asociada a la base de datos desarrollada para el proyecto descrito en este documento. El diagrama ha sido realizado con el plugin del IDE Eclipse: ER-Master [ERM].

.

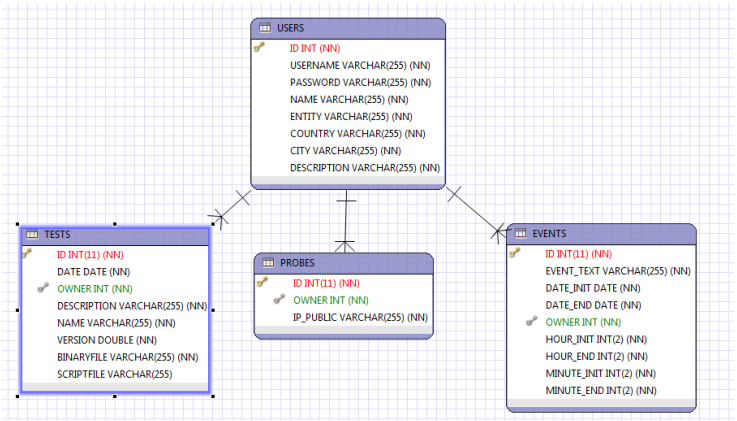

Ilustración 9. Diagrama E/R de la base de datos

3.3.4 Arquitectura: J2EE

Java 2 Plattform Enterprise Edition [J2EE] es una plataforma de programación Java perteneciente la empresa Oracle que provee una API y un entorno de ejecución para el desarrollo y ejecución de software empresarial, incluyendo servicios web y otras aplicaciones de red a gran escala, multi-nivel, escalable, confiable y seguro. J2EE es una extensión de JSE, proveyendo una API para el mapeo objeto-relacional, distribuido, arquitecturas multi-nivel, y servicios web. El diseño de J2EE es modular y el software desarrollado en él principalmente se realiza con el lenguaje de programación Java.

J2EE incluye varias especificaciones de APIs, como JDBC, XML, etc. y define como coordinarlos. J2EE también incluye algunas especificaciones únicas para los componentes de Java EE. Estos incluyen Enterprise JavaBeans, Conectores, servlets, JSPs, etc. Esto permite a los desarrolladores crear aplicaciones empresariales que son portables y escalables, y que se integran con las tecnologías existentes.

Para el desarrollo de este proyecto se ha buscado la descomposición modular, siguiendo el modelo "divide y vencerás". Así podemos descomponer la aplicación web en los siguientes módulos:

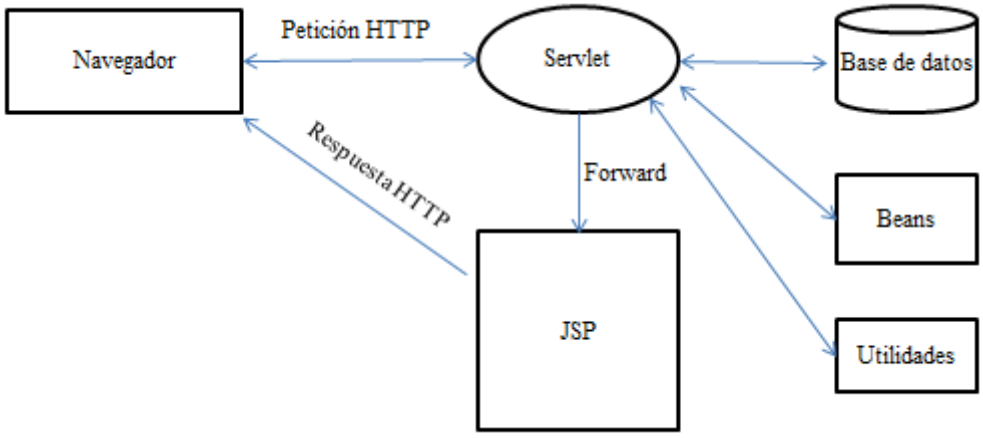

Ilustración 10. Módulos de la aplicación web

- JSPs: Ficheros estáticos que contienen el código HTML que verá el usuario en su navegador. El uso de JSPs permite la utilización de código Java incrustado sobre el código HTML facilitando ciertas operaciones.
    - "*MyTests.jsp*"
    - "*Tests.jsp*"
    - "*DeleteTest.jsp*"
    - (…)
- Servlets: Recogen la petición web, realizan las gestiones oportunas (autenticación, interactuar con la base de datos, etc.) y realizan redirecciones a ficheros JSPs.
    - "*Tests.java*"
    - "*DeleteTest.java*"
    - "*Authentication.java*"
    - (…)

- Beans: Clases Java que definen objetos. Entre ellas podemos encontrar:
    - "*Event.java*" – atributos y métodos relacionados con eventos.
    - "*Planner.java*" – atributos y métodos relacionados con los planificadores.
    - "*Probe.java*" – atributos y elementos que definen una probe.
    - "*Test.java*" – atributos y elementos que definen un test.
    - "*User.java*" - atributos y elementos que definen un usuario.
- Utilidades: Se trata de clases Java con funciones útiles para la realización de objetivos específicos. Un ejemplo sería la clase "*Base.java*", donde están albergados todos los métodos correspondientes a la interacción con la base de datos o la clase "*SSHUtils*", donde podemos encontrar los diferentes métodos usados por los Servlets relativos al uso de SSH.
    - "*Base.java*"
    - "*FileUtils.java*"
    - "*JsonUtils.java*"
    - "*NullHostKeyVerifier.java*"
    - "*SSHUtils.java*"

3.3.5 Conexión SSH: SSHJ

SSHJ es una librería para Java que permite la conexión vía SSH entre dos máquinas a través del código [SSHJ]. Las características de la librería incluyen:

- Lectura de ficheros "*known_hosts*" para la verificación de claves de host.
- Autenticación por: publickey, contraseña y teclado interactivo.
- Canales de comando, subsistema y Shell.
- Reenvío de puertos local y remoto.
- Implementación de SCP y FTP.

Los requerimientos de la librería son mínimos: Java versión 6 o superior y slf4 [SLF4J] son imprescindibles. Bouncycastle [BOUC] es altamente recomendado y necesario para usar algunos de los algoritmos de cifrado. Jzlib [JZL] es necesario para el uso de la compresión zlib [ZLIB] .

Debido a todo lo explicado, se decide usar esta librería para realizar las conexiones entre la aplicación web y el nodo controlador. Para usar la librería basta con añadirla al proyecto y cumplir los requerimientos explicados previamente. A continuación se puede observar un ejemplo del uso de esta librería. Se trata de la función encargada de conectar y lanzar un comando en una máquina remota:

```
public void executeCommand(String command){
    Command cmd = null;
    System.out.println("Vamos a ejecutar un comando por SSH: "+command);

    //Llamada a la función encargada de realizar la conexión SSH
    SSHClient ssh = connect();
    try{

        //Comienza la sesión
        final Session session = ssh.startSession();

        //Se lanza el comando recibido por parámetro
        cmd = session.exec(command);

        //Se imprime por pantalla el resultado de la ejecución
        System.out.println("Resultado:");
        System.out.println(IOUtils.readFully(cmd.getInputStream()).toString());
    } catch(IOException e){
        e.printStackTrace();
    }
    disconnect(ssh);
}
```

Ilustración 11. Ejemplo de SSHJ

3.3.6 Calendario: fullCalendar

FullCalendar [FULLC] es un plugin de jQuery [JQ] que provee un calendario de tamaño completo con funciones de arrastrar y soltar. Usa AJAX [AJAX] para traer eventos al vuelo para cada mes y es fácilmente configurable para usar el formato que uno desee. Se puede personalizar visualmente, y expone los ganchos para los eventos activados por el usuario (por ejemplo, pusar un elemento o arrastrar un evento) y tiene licencia de código abierto bajo la licencia MIT.

FullCalendar es ideal para la visualización de eventos sobre el calendario, pero no proporciona en sí una solución completa para la gestión de contenidos. Para ello se puede apoyar uno en las librerías de AJAX o JSON [JSON].

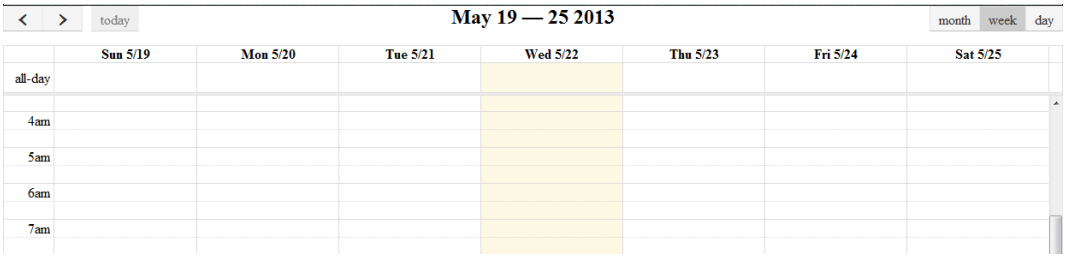

Ilustración 12. Ejemplo de fullCalendar

3.3.7 Herramientas de desarrollo

Se han podido observar los diferentes componentes que forman parte del desarrollo de la aplicación web. Se deben destacar dos herramientas de gran utilidad usadas a lo largo del desarrollo: XAMPP y Eclipse.

XAMPP es un compendio gratuito de diferentes herramientas que ayudan al desarrollo web [XAMPP]. Se trata de una forma fácil de realizar una instalación de Apache y contiene: Apache, MySQL, PHP + PEAR, Strawberry Perl , mod_php, mod_perl, mod_ssl, OpenSSL, phpMyAdmin, WebAlizer, Mercury Mail Transport System, NetWare System, JpGraph, FileZilla, FTP Server, mcrypt, eAccelerator, SQLite, WEB-DAV, Tomcat, y un completo panel de control de cada uno de sus componentes. Esta herramienta además está disponible para distribuciones Linux, Windows, Mac OS X y Solaris.

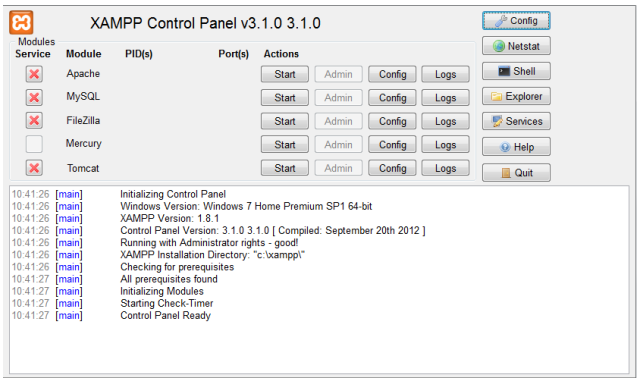

Ilustración 13. Panel de control de XAMPP

Por otro lado, el entorno de desarrollo utilizado en este proyecto es la plataforma Eclipse [ECL]. Se trata de un entorno de desarrollo integrado (IDE) ya que provee herramientas para gestionar espacios de trabajo; construir, lanzar y realizar debug de aplicaciones; y para personalizar fácilmente la experiencia de programación [ECLb].

Eclipse es uno de los entornos de desarrollo más extendidos a lo largo del sector tecnológico. Su facilidad de uso, gratuidad y su gran variedad de extensiones (plugins) hacen de este IDE la elección perfecta para llevar a cabo el desarrollo de un proyecto como el descrito en estas páginas.

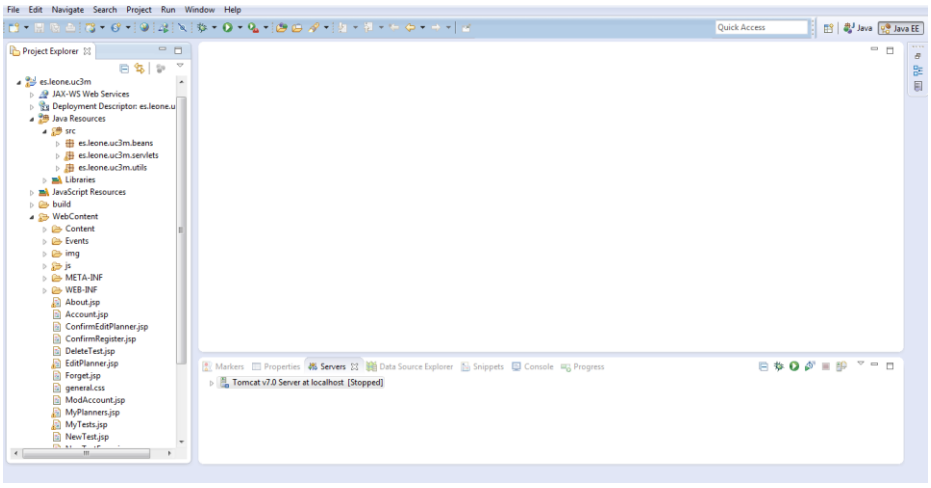

Ilustración 14. Eclipse

3.3.8 Plataforma medidas de Samknows

A continuación se va a detallar el funcionamiento general de la plataforma de medidas a gran distancia desarrollada e implementada por Samknows.

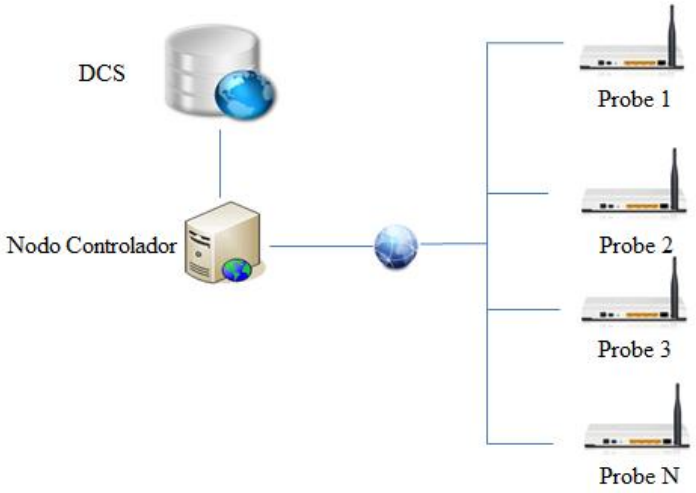

Ilustración 15. Plataforma de medidas de Samknows

Como podemos observar en la figura anterior la infraestructura desarrollada por el socio Samknows se basa en el uso de varios elementos: probes, servidores, nodo controlador y base de datos. Las probes repartidas por la red realizan tests contra diferentes servidores tomando diferente tipo de medidas ("*http_get*", "*jitter*", etc.). Los resultados de estos tests son enviados a un nodo controlador, el cual, además de gestionar todo el sistema de actualización del contenido de las probes y su autenticación, guarda los resultados en una base de datos propia. Los resultados pueden ser visibles, a través de una interfaz amigable, tanto por los usuarios finales que albergan las probes en sus redes hogar, como por los diferentes socios del proyecto Leone.

Todas las probes son actualizadas de forma periódica preguntando al nodo controlador cada hora si hay actualizaciones disponibles (utilizando un mecanismo de autenticación). Las actualizaciones de las probes se distribuyen a través de paquetes comprimidos con diferente tipo información: planificación, binarios tests, scripts, etc. Estos paquetes de actualización son construidos por el nodo controlador, en concreto por el script "*packager/build.php*". Antes de ser enviados a las diferentes probes con el contenido, el paquete es almacenado localmente por el nodo controlador en la ruta "*/opt/Samknows/ispmon/fcc/DCS/packages*".

El contenido de los paquetes de actualización, se nutre del contenido de ficheros alojados en diferentes carpetas pro el nodo controlador. A continuación, se van a detallar las diferencias entre las cuatro carpetas relacionadas con el contenido de las actualizaciones de las unidades:

1. **Panel:** contiene carpetas con nombres de diferentes proyectos, por ejemplo, FCC ó UK –Ofcom-. Estos proyectos, pueden contener múltiples unidades a lo largo de múltiples ISPs con múltiples tipos de hardware. Cualquier archivo de configuración que se desee utilizar para un proyecto concreto debería ir aquí. Para el proyecto descrito en este documento se va a utilizar el panel "*leone*", el cual engloba todas las unidades relativas al proyecto Leone.
2. **Base:** contiene ficheros de configuración adaptados para los diferentes tipos de hardware que puede soportar la infraestructura. En el caso del proyecto presente, se utilizará la carpeta "*wr741nd*" correspondiente al hardware utilizado a lo largo de éste.
3. **Skel:** contiene ficheros de configuración específicos para los ISPs. Por ejemplo, se puede tener los skel "*leone*" (con la configuración estándar) y "*leone-double*" (lanzando tests el doble de veces de lo estándar). Con estos skel, se podría asignar el skel "*leone-double*" a algunas unidades consiguiendo, de esta manera, que las unidades cogiesen su configuración del directorio "*leone-double*". Se trata de una alternativa para configurar varias unidades sin necesidad de utilizar la carpeta "*unit*".
4. **Unit:** contiene ficheros de configuración para unidades específicas. Todos los ficheros almacenados aquí, sobrescribirán lo que exista en los directorios "*panel/*", "*base/*" y "*skel/*". Por ejemplo, "*/opt/samknows/ispmon/fcc/packager/unit/12345*" contendría la configuración específica de la unidad 12345 y el paquete de actualización se realizaría así: "*/opt/samknows/ispmon/fcc/packager/build.php 12345*". Si se deseara realizar cambios para todas las unidades, se debería hacerlo en las carpetas "*panel/*", "*base/*" o "*skel/*".

Las probes tienen gestionado un funcionamiento de forma genérica y periódica a través del fichero "*crontab*". En él se indica qué acciones acomete y el instante en el que debe realizarlas: autenticación y descarga de actualizaciones con el nodo controlador, ejecutar unos tests genéricos de forma periódica, eliminar contenido de pruebas obsoleto, enviar resultados de los tests al DCS, sincronización de tiempo, etc.

Así, las unidades, por defecto, tienen planificado el lanzamiento, cada 2 horas, de los tests: *HTTP_GET*, *HTTP_POST*, *UDP_Jitter*, *Webget*. La planificación se construye por el nodo controlador, para todas las tests, en el paso 10 del fichero de construcción de paquetes de actualización. En la planificación también intervienen los ficheros "*crontab*" y "*jobfile*", en los cuales se profundizará más adelante.

Antes de lanzar cualquier test, la unidad realiza una serie de comprobaciones, por ejemplo, comprueba si se supera el umbral de cantidad de tráfico que tiene por sus interfaces. Para estas comprobaciones se apoya en diferentes ficheros, como por ejemplo, "*simple_thresh_checker.sh*". Este fichero en concreto, realiza la comprobación citada anteriormente y, si se cumplen los requisitos, lanza en serie los tests indicados por el fichero "*jobfile*". En caso de no superar la comprobación, retrasa la ejecución de los tests hasta que las condiciones sean favorables para su ejecución. Además antes de ejecutar cada test se debe para el proceso encargado de capturar paquetes y después de haber terminado la ejecución de cada uno se debe volver a poner en marcha.

3.3.9 Caso de uso 1 – Insertar un nuevo test

A continuación se va a realizar un estudio minucioso sobre cómo copiar un nuevo test en todas las unidades. Un test está compuesto por, al menos, un archivo binario almacenado en la carpeta "*bin*" y un archivo script almacenado en la carpeta "*scripts*". Estos tests, pueden realizar cualquier tipo de medida dentro de la infraestructura y luego enviar los resultados al DCS para almacenarlos con el resto de medidas tomadas a lo largo de las pruebas efectuadas.

El envío y almacenaje de los ficheros de cada test está apoyado del sistema de actualizaciones de la infraestructura desplegada por el socio Samknows. A continuación se va a explicar, paso por paso, uno de los procedimientos para poder alojar los ficheros de un test en todas las probes de la infraestructura:

- Las ficheros de actualización pertenecientes a todas las unidades del proyecto Leone se almacenan en el directorio del controlador "*/opt/samknows/ispmon/fcc/packager/panel/*".
    - Fichero binario:
        - Debe tener un nombre único.
        - Se almacenará en el directorio "*panel/leone/ispmon/bin/*" del nodo controlador.
    - Fichero script:
        - Debe incluir todas las variables que pueda necesitar el test a la hora de ser ejecutado.
        - Se almacenará en el directorio "*panel/leone/ispmon/scripts/*" del nodo controlador.
- En segundo lugar, se debe construir un paquete de actualización para todas las probes. Existen dos maneras de realizar este tercer paso:
    - Opción 1: realizando un paquete para cada unidad individualmente:
        - "*Php –q build.php <ID o MAC>*"
    - Opción 2: realizando un paquete para todas las unidades simultáneamente:
        - "*Build-all-leone.sh*"
- Las probes recibirán el paquete de actualización en la próxima "llamada a casa" al nodo controlador. Una vez recibido, almacenarán el archivo binario en la carpeta "*bin*" y el fichero script en la carpeta "*scripts*".

Si algún usuario deseara lanzar el test, sería tan sencillo como realizar la llamada correctamente al script de control de dicho test.

Los resultados de los tests deben ser enviados al DCS, donde se almacenan el resto de resultados de los tests. Éstos deben tener un formato específico (el cual se puede consultar en el fichero "*/opt/Samknows/ispmon/fcc/DCS/config.properties*" del nodo controlador). Es imprescindible tener una tabla por métrica/test.

3.3.10 Caso de uso 2 – Planificación de tests

A continuación se va a detallar el proceso de planificación que sufre un test desde que un usuario desea planificar su ejecución hasta que la ejecución ha sido llevada a cabo.

Las unidades de la infraestructura de Samknows tienen una planificación de ejecución de una serie de tests determinados de forma periódica. El detalle de esa planificación específica, para todas las probes, se realiza en el proceso de construcción del paquete de actualización (con el fichero "*build.php*", el cual se encuentra alojado en el servidor central).

Actualmente existen dos maneras de realizar una planificación para un test a través de dos ficheros concretos: "*jobfile*" y "*crontab*". El primero contiene un listado con un conjunto de tests que serán ejecutados en serie. La llamada a este fichero la realiza el fichero "*simple_thresh_checker*", que a su vez es llamado por el fichero "*pcscript*", que está planificado para ser ejecutado todas las horas del día en el minuto 19 de cada hora. Por otro lado, tenemos el segundo fichero, "*crontab*", el cual planifica de forma exacta la planificación por minutos y horas de diferentes archivos del sistema. Por ejemplo, como se citaba anteriormente, la ejecución del script "*pcscript*" cada hora en el minuto 19.

Los dos ficheros de actualización son almacenados por el nodo controlador y se entregan a las unidades dentro del paquete de actualización. Como se pudo observar anteriormente, existen diferentes carpetas en las que pueden ser almacenados dichos archivos y dependerá de donde se ubique un fichero para que sea o no sobrescrito. Por ejemplo, si tenemos una planificación en un fichero "*crontab*" en el "*/opt/samknows/ispmon/fcc/packager/panel/ispmon/cron/crontab*" y otro en la "*unit/<UNIT-ID>*", el segundo fichero sobrescribirá el contenido del primero para el paquete de actualización de la unidad con el identificador correspondiente.

Si se opta por incluir el test en el fichero "*jobfile*", éste se ejecutará en serie con el resto de tests listados en ese fichero. Además, el sistema realizará las comprobaciones necesarias de forma automática (comprobación de umbral de ancho de banda, parada de captura de paquetes y reanudación al finalizar su ejecución así como el envío de los resultados a través del script "*submit.sh*").

Si por el contrario se opta por utilizar el fichero de planificación "*crontab*", será necesario añadir la línea correspondiente a la llamada del script indicando el instante en el que se desea que sea ejecutado. Un ejemplo de una línea de este fichero sería: "*19 * * * * /tmp/ispmon/scripts/pcscript >/dev/null 2>&1*". En este caso, el sistema no realizaría las comprobaciones necesarias automáticamente, sino que el script de control llamado debería realizar la comprobación del umbral del ancho de banda, el parado y reanudación del proceso de captura de paquetes así como el envío de resultados del test.

En el segundo caso además se añade la desventaja de que dos tests no pueden ser ejecutados al mismo tiempo. Esto podría resolverse, de una manera sencilla, comprobando que el proceso "*pcscript*" no está en ejecución en la probe.

En la planificación de tests juega un papel importante también el script "*shouldrun.sh*", implementado por Samknows. Este script, comprueba si un test puede ser ejecutado a una hora determinada. Los ficheros con las horas están almacenados en "*/tmp/ispmon/Schedule/"* y solo contienen enteros, por ejemplo, el contenido podría ser: "*0 2*

4 6". Pasándole el nombre del fichero almacenado en la carpeta "*Schedule*", "*shouldrun*" comprueba si, en ese momento, nos ubicamos en alguna de las horas permitidas para su ejecución.

A continuación se indica el procedimiento de un caso concreto para intentar resolver las dudas que puedan haber surgido. Suponiendo que se deseara realizar una planificación para lanzar un test (leonetest) de forma independiente, en una sola unidad utilizando el fichero crontab podría realizarse siguiendo el siguiente procedimiento:

- Ubicar en la carpeta:
  - *cd /opt/samknows/ispmon/fcc/packager/unit*
- Crear las carpetas "*ispmon*" y "*cron"* dentro de la carpeta del test:
  - *mkdir -p leonetest/ispmon/cron*
- Copiar el contenido del archivo "*crontab*" en la subcarpeta "*cron*" de la carpeta del test:
  - *cp -p ../panel/leone/ispmon/cron/crontab leonetest/ispmon/cron*
- Añadir la línea del fichero "*crontab*" para la planificación del test al nuevo crontab:
  - *echo <crontab line for the test schedule> >> leonetest/ispmon/cron/crontab*
- Crear un enlace simbólico:
  - *ln -s $ID leonetest*
- Por último bastaría con crear el paquete de actualización referente a esa unidad:
  - */opt/samknows/ispmon/fcc/packager/build.php $ID*

Este proceso tiene el siguiente inconveniente; si se quiere posteriormente "expulsar" otro test de alguna de las probes (pero no de todas), los enlaces simbólicos utilizados causarán problemas. Para solventarlo, se debería crear una carpeta debajo de "*unit*" para cada probe en lugar de usar enlaces simbólicos. De esta manera, se puede asegurar que cada unidad tiene su configuración individual y, en ese caso, solo sería necesario actualizar el "*crontab*" de la unidad pertinente.

Para garantizar la seguridad de recuperación en caso de fallo, se debe mantener un "*crontab*" por defecto, por ejemplo el ubicado en "*panel/leone*". De esta manera cada vez que se desee restaurar una probe a su contenido por defecto se debería copiar ese "*crontab*".

3.3.11 Caso de uso 3 – Borrar un test existente

El caso de uso que se va a analizar a continuación consiste en la eliminación de un test en todas las unidades, previamente ubicado en todas ellas. Se ha pensado en la posibilidad de que, pasado un cierto tiempo, el desarrollador de un test considere que éste ha quedado obsoleto y desee eliminarlo de todas las unidades, para lo cual simplemente se debería eliminar los archivos pertinentes a ese test en el nodo controlador ("*panel/ispmon/etc/{scripts,bin}*" o "*unit/<id>/ispmon/etc/{scripts,bin}*") y después lanzar el paquete de actualización para todas las unidades.

Este caso tiene una serie de problemas relacionados con la planificación previa de dicho test. Si no se modifican los archivos pertinentes a su planificación, en caso de que este se haya producido previamente, la probe lanzaría mensajes de error (relevantes a no encontrar los ficheros) y continuaría con el resto de la planificación. Para solventar estos problemas se plantean varias posibilidades:

- Realizar un sistema de análisis sintáctico para comprobar el contenido de los ficheros de planificación y eliminar la planificación específica de dicho test.
- Notificar a los usuarios del borrado de este test para que procedan con el borrado de su planificación de forma manual.

La resolución de estos problemas, así como el desarrollo de cualquiera de sus posibles implementaciones quedan más allá del alcance de este proyecto, quedando así para un futuro trabajo.

Capítulo 4

Pruebas y resultados

4.1 Introducción

A lo largo de este capítulo se detallará el desarrollo de la fase de pruebas del proyecto. El proceso de pruebas constituye la herramienta adecuada para determinar el estatus de la calidad del producto de software desarrollado. A lo largo de esta fase, se ejecutan pruebas enfocadas a los diferentes componentes del sistema o al sistema de software en su totalidad, con el principal objetivo de medir el grado con el que el software desarrollado cumple con los requisitos deseados.

Existe una clasificación de las diferentes fases que recorren las pruebas de software ordenadas de forma ascendente en función del nivel: pruebas de unidad, pruebas de integración, pruebas de validación, o aceptación, y pruebas del sistema. A continuación se detallan las características de cada una de las fases.

Las pruebas de unidad tienen por objetivo validar cada una de las unidades, o componentes, del sistema de forma aislada. Para realizar correctamente esta primera fase de pruebas, se deben realizar pruebas sobre todas las unidades o componentes del sistema de forma aislada, comprobando su correcta funcionalidad en entornos controlados y aislados del resto del sistema. En este documento detallarán cuatro pruebas, escogidas de forma aleatoria del conjunto de pruebas unitarias del sistema, donde se pretende dar una idea del proceso de pruebas elaboradas a lo largo de esta primera fase.

Las pruebas de integración tienen por objetivo validar la interacción entre varias unidades o componentes del sistema. Para su correcta validación, deben ser probadas todas las relaciones posibles entre las unidades o componentes del sistema completo. En este documento, se describe una prueba del conjunto de pruebas realizadas en esta segunda fase. Esta prueba ha sido escogida de forma aleatoria.

Las pruebas de validación o aceptación tienen por objetivo comprobar que el sistema cumple con los requisitos deseados desde el punto de vista de la funcionalidad y del rendimiento. Esta fase de pruebas debe demostrar que el producto final cumple con los requisitos y expectativas acordadas. En este documento, se detallará un caso de prueba del conjunto de pruebas de validación realizadas al sistema. El caso de prueba ha sido escogido como ampliación del caso descrito en la segunda fase del proceso de validación del proyecto.

Las pruebas del sistema constituyen la última fase de pruebas y tienen por objetivo, validar el sistema completo en todos los ámbitos de interés: estructura, funcionalidad, comunicación, carga, recuperación, accesibilidad, usabilidad, seguridad y operación. En este documento se va a describir una prueba, del conjunto de pruebas realizadas al sistema completo, en concreto, se ha escogido una prueba del conjunto de pruebas funcionales. A través de un caso de uso del sistema se pretende demostrar cómo se han realizado estas pruebas del sistema completo.

4.2 Pruebas de unidad

A continuación se enumerarán y describirán las pruebas referentes el testeo de las unidades o componentes de forma aislada [SCRa]. El fin de estas pruebas es intentar encontrar defectos de cada componente de forma independiente e individual. Se consideran unidades o componentes: funciones individuales, métodos dentro de un objeto, instancias de clases con varios atributos y métodos, o componentes, compuestos con interfaces usadas para acceder a su funcionalidad.

Las pruebas de unidad involucran todas las operaciones asociadas a un objeto, uso e instanciación de todos los atributos del objeto, pruebas del objeto en todos los estados posibles e identificación de transiciones (incluyendo las secuencias de eventos que causan estas transiciones).

Para llevar a cabo las pruebas unitarias se han realizado pruebas sobre todos y cada una de las unidades comprobando su correcto funcionamiento de forma aislada. Por motivos de espacio no se incluyen en este documento todas las pruebas realizadas. Para ejemplificar ese proceso de pruebas, a continuación, se detallan cuatro pruebas realizadas a cuatro métodos implementados, siendo éstos elegidos de forma aleatoria. Para ello se utiliza una de las clases auxiliares, Base.java, la cual está encargada de la gestión de la base de datos. A continuación se detallan las diferentes pruebas realizadas de forma independiente:

1. **Unidad de prueba nº1:** Método searchUsuario().
2. **Unidad de prueba nº2:** Método insertNewTest().
3. **Unidad de prueba nº3:** Método getTestByName().
4. **Unidad de prueba nº4:** Método deleteTest().

4.2.1 Medida de la unidad de prueba nº1

Método	searchUsuario()
Objetivo	Buscar un usuario en la base de datos
Entrada	Nombre utilizado como nombre de usuario
Pruebas	Se crea un usuario manualmente en la base de datos y luego se llama a este método pasándole por parámetro el nombre de usuario
Salidas	True, ya que el usuario se encuentra en la base de datos almacenado correctamente
Evaluación	El resultado es el esperado

Tabla 1. Prueba unitaria nº1

4.2.2 Medida de la unidad de prueba nº2

Método	insertNewTest()
Objetivo	Insertar un nuevo test en la base de datos
Entrada	Nombre de usuario, nombre del test, descripción del test, versión, nombre del fichero binario, nombre del fichero script
Pruebas	Se crean dos tests con el mismo nombre de test
Salidas	False, ya que no pueden existir dos tests en la base de datos con el mismo nombre
Evaluación	El resultado es el esperado

Tabla 2. Prueba unitaria nº2

4.2.3 Medida de la unidad de prueba nº3

Método	getTestByName()
Objetivo	Obtener un test de la base de datos dado su nombre
Entrada	Nombre del test
Pruebas	Se crea un test y se inserta en la base de datos. Luego llamamos a este método para comprobar que podemos obtenerlo
Salidas	True, ya que al haberlo almacenado previamente de forma correcta podemos obtenerlo
Evaluación	El resultado es el esperado

Tabla 3. Prueba unitaria nº3

4.2.4 Medida de la unidad de prueba nº4

Método	deleteTest()
Objetivo	Eliminar un test de la base de datos
Entrada	Nombre del test
Pruebas	Se crea un test pero, antes de llegar a almacenarlo, se intenta eliminar de la base de datos
Salidas	False, ya que al no haber sido almacenado en la base de datos es imposible borrarlo
Evaluación	El resultado es el esperado

Tabla 4. Prueba unitaria nº4

Para realizar este proceso de pruebas se llevaron a cabo pruebas como las citadas anteriormente sobre todas las unidades de todos los módulos. Los resultados de dichas pruebas fueron satisfactorios demostrando que la implementación de dichas unidades fue realizada de manera correcta.

4.3 Pruebas de integración

Una vez finalizadas todas las pruebas unitarias del proyecto verificando el correcto funcionamiento de cada uno de los componentes del mismo de forma aislada, se procede a realizar las pruebas de integración [SMpi]. Éstas consisten en construir el sistema a partir de sus componentes y realizar pruebas buscando problemas que puedan surgir en la interacción entre sus componentes [Integ].

Existen diferentes metodologías para la realización de este tipo de pruebas: Integración Ascendente, Integración Descendente, Pruebas de Regresión y Pruebas de Integración Incremental. En el caso específico de software de objetos, cuando se usa algún método derivado de casos de uso, éstos brindan una forma práctica de realizar pruebas de integración, guiadas por los diagramas de colaboración, o de secuencia, asociados con cada caso de uso.

Al desarrollar software orientado a objetos guiado por casos de uso, éstos forman las unidades mínimas de funcionalidad [Fer10]. Para la elaboración de las pruebas se toman diferentes diagramas de secuencia con uno o varios casos de prueba por cada diagrama. En un diagrama de secuencias se muestran una serie de objetos, representando clases que forman parte del sistema o elementos almacenados en disco (archivos, bases de datos, etc.). De cada elemento será de interés su estado (existe, no existe, existe pero con versión inadecuada, no accesible, listo, desconectado o en problemas).

A continuación se detalla uno de los casos de prueba desarrollados a lo largo del proceso de pruebas de integración. En concreto, se trata de una prueba referente a la parte de autenticación de usuario de la aplicación web. En la siguiente figura se muestran las clases que participan en este caso de uso. Una clase actúa como frontera (Servlet), otra como control, encargándose de la base de datos, y una última clase para mostrar los resultados al usuario (JSP).

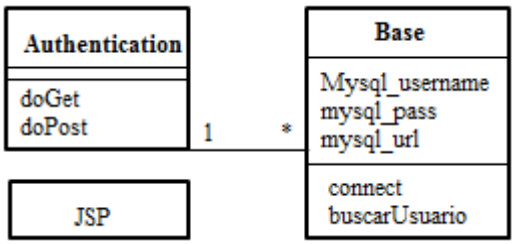

Ilustración 16. Prueba de integración - Diagrama de clases

La siguiente figura muestra un diagrama de secuencia para el escenario en el que un usuario se identifica satisfactoriamente.

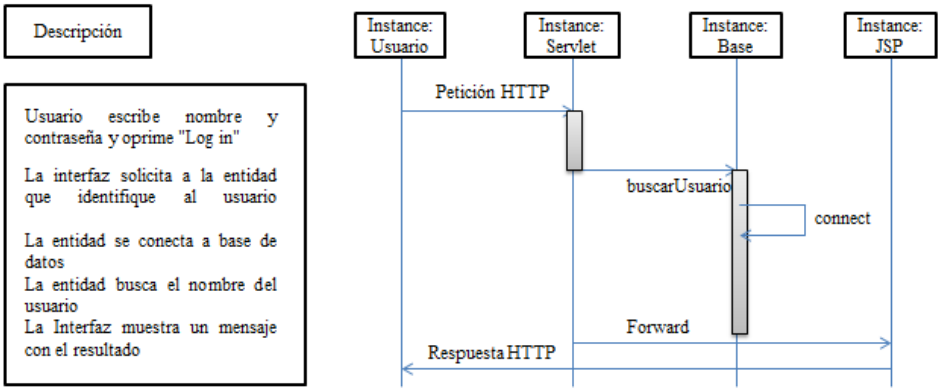

Ilustración 17. Prueba de integración – Diagrama de secuencia de caso exitoso

El resultado de esta prueba fue satisfactorio cumpliendo con el objetivo esperado. Para simplificar el documento no se incluirá todo el código Java correspondiente a este caso de uso. Tras realizar todas las pruebas de integración de todos los componentes del sistema, se observa como los resultados cumplen los esperados, permitiendo la validación del sistema a través de las pruebas de integración.

4.4 Pruebas de validación

A continuación se procede con el tercer paso en la realización de pruebas de software: pruebas de validación, también llamadas, pruebas de aceptación [SMPA]. Éstas tienen por objetivo, validar que el sistema cumple con los requisitos desde el punto de vista de la funcionalidad y del rendimiento. A partir de dichas pruebas, se debe demostrar que el producto final cumple con los requisitos y expectativas acordadas.

Por motivos de espacio no se incluyen todas las pruebas de validación realizadas al sistema. Para mostrar un ejemplo de este conjunto de pruebas y, continuando con el caso de prueba descrito en la fase anterior, a continuación se detallan las pruebas realizadas en el caso de uso de la autenticación de un usuario. En primer lugar se puede observar la tabla con los diferentes estados posibles de todos los elementos de la prueba:

Elemento	Estados
Servlet (Authentication)	Presente, ausente
JSP (TittlePage)	Presente, ausente
Clase controladora (Base)	Presente, ausente
Base de datos	Activa, con problemas

Tabla 5. Prueba de validación – Estados

Partiendo de que el usuario solo necesita introducir dos parámetros para el acceso (usuario y contraseña) y éstos son entregados, sin realizar cambios, a la clase controladora y de ahí a la base de datos; sólo se deben considerar las dos variables. A continuación se puede observar en la tabla el conjunto de valores posibles de los parámetros de entrada:

test_user2, 1a2b3c4@	Nombre y contraseñas registradas
test_user2, 1234	Nombre registrado, contraseña mal
test_user2	Falta contraseña
, 1a2b3c4@	Falta nombre
user_test, 1234	Nombre no registrado

Tabla 6. Prueba de validación – Valores de entrada

Combinando las dos tablas anteriormente descritas, se forman los casos de prueba que se incluyen en la siguiente tabla. Se puede observar cómo los estados se reflejan en las condiciones de entrada.

Entrada	Condiciones de entrada	Salida esperada
test_user2, 1a2b3c4@	Servlet (Authentication) ausente	Fallo de ejecución
test_user2, 1a2b3c4@	Clase Controladora (Base) ausente	Falo de ejecución
test_user2, 1a2b3c4@	Base de datos con problema	Fallo de ejecución
test_user2, 1a2b3c4@	JSP (TittlePage) ausente	Fallo de ejecución
test_user2, 1a2b3c4@	Servlet (Authentication) presente Clase Controladora (Base) presente Base de datos activa	Mensaje de bienvenida
test_user2, 1234	Servlet (Authentication) presente Clase Controladora (Base) presente Base de datos activa	Mensaje de error en contraseña
test_user2	Servlet (Authentication) presente Clase Controladora (Base) presente Base de datos activa	Mensaje de error en contraseña

, 1a2b3c4@	Servlet (Authentication) presente Clase Controladora (Base) presente Base de datos activa	Mensaje de error en nombre
user_test, 1234	Servlet (Authentication) presente Clase Controladora (Base) presente Base de datos activa	Mensaje de error en nombre

Tabla 7. Prueba de validación – Casos de prueba

Tras ejecutar el conjunto de pruebas detallado anteriormente, se obtienen los resultados esperados. Además de esta prueba concreta, se realizaron pruebas de validación para todos los componentes del sistema y, tras obtener los resultados deseados, se valida el sistema desde el punto de vista de pruebas de validación o aceptación.

4.5 Pruebas del sistema

Por último, se van a realizar las pruebas referentes a las denominadas pruebas de sistema [SMPS]. Estas pruebas tienen por objetivo validar el sistema completo, usando para ello pruebas específicas en cada uno de los diferentes ámbitos posibles:

- **Pruebas de Estructura**: son pruebas de caja blanca para comprobar las estructuras de los datos internos.
- **Pruebas Funcionales**: determinan si el sistema completo hace lo que se requiere de él. Ignoran los mecanismos internos de un sistema o componente centrándose en las salidas y respuestas del sistema en base a los requisitos del mismo. Se denominan pruebas de caja negra.
- **Pruebas de Comunicación**: detectan posibles problemas de comunicación de la aplicación con otros sistemas.
- **Pruebas de Carga**: permiten conocer el comportamiento al trabajar con grandes volúmenes de usuarios y/o datos.
- **Pruebas de Recuperación**: comprueban si la aplicación es capaz de recuperarse ante un fallo sin comprometer la integridad de los datos.
- **Pruebas de Accesibilidad**: comprueban si es posible acceder al sistema desde los entornos para los que ha sido diseñado.
- **Pruebas de Usabilidad**: grado de satisfacción y facilidad de uso de los diferentes interfaces de usuario de la aplicación para sus usuarios objetivo.
- **Pruebas de Seguridad**: comprueban si los mecanismos de seguridad evitan alteraciones indebidas en los datos.
- **Pruebas de Operación**: posibilidad de realizar las diferentes operaciones de mantenimiento (arranques, paradas, copias de seguridad, etc.) sin comprometer el nivel de servicio esperado para la aplicación.

Por motivos de espacio, a continuación solo se va a detallar un caso de prueba de las llamadas pruebas de funcionalidad. En concreto, el siguiente caso de prueba consiste en la realización de uno de los casos de uso principales del proyecto descrito a lo largo de este documento: carga de un nuevo test por parte de un usuario. A continuación se completan todos los detalles de la prueba:

Resumen:	Este caso de prueba describe como realizar el proceso de carga de un nuevo test a todas las probes de la plataforma de medidas del proyecto Leone.
Precondiciones:	• Acceso a la aplicación web. • Nombre de usuario y contraseña para poder acceder al entorno de pruebas.
Postcondiciones:	• Realizar correctamente la carga del nuevo test.

Tabla 8. Prueba de sistema

Pasos:

1. Acceder a la aplicación web introduciendo su URL <*url*> en el navegador.
2. Introducir nombre de usuario <*usuario*> y contraseña <*contraseña*> y pulsar el botón "Log in".
3. Pulsar el botón "My Tests".
4. Pulsar el botón "Create new test".
5. Introducir los datos del test: <*nombre*> y <*descripción*>. Seleccionar el archivo binario, y el fichero script, del test pulsando el botón "Examinar…".Pulsar el botón Validate.

Resultados Esperados:

1. Verificar que se ha accedido correctamente a la URL <*url*>. Verificar que se muestran los controles para realizar el login mediante usuario y contraseña.
2. Verificar que se muestra el mensaje de bienvenida "Hi, <*nombre_usuario*>".
3. Verificar que se muestra el título "My Tests".
4. Verificar que se muestran los campos del formulario "Name", "Description", "Binary" y "Script".
5. Verificar que se muestra el título "My Tests". Verificar que se muestra <*nombre*> en la columna "Name". Verificar que se muestra <*descripción*> en la columna "Description". NOTA: si ha ocurrido algún fallo en la validación del formulario se mostrará un popup indicando el error.

Tras realizar el caso de prueba descrito y verificar que los resultados se ajustan con los esperados, se valida y cierra el caso. Además de este caso se han realizado pruebas para validar los diferentes aspectos que componen todas las pruebas de sistema. Tras comprobar que los resultados cumplen con lo esperado, se da por finalizado la fase de pruebas del proyecto.

Capítulo 5

Planificación y presupuesto

5.1 Introducción

A lo largo de este capítulo, se detalla, con la máxima precisión posible, las tareas llevadas a cabo a lo largo del proyecto, así como su planificación y el desglose del presupuesto final del mismo.

En primer lugar, se incluye la planificación llevada a cabo para la elaboración del proyecto. En este apartado, se describen las diferentes fases de las que consta el proyecto, incluyendo detalles precisos de cada una, como, por ejemplo, la fecha de inicio y finalización de cada una de las actividades. Al final del apartado de planificación, se muestra un diagrama representativo de la secuenciación de dichas fases. Para la elaboración del diagrama, se ha utilizado una herramienta muy extendida en proyectos de ingeniería llamada diagrama de Gantt, por varias razones, destacando su gran utilización por el conjunto del sector ingenieril, así como por su gran sencillez en la visualización de la secuenciación de las tareas.

En la segunda parte de este capítulo, se desglosa, de forma exhaustiva, el presupuesto total del proyecto derivado del conjunto de gastos totales del mismo. Para la realización de este presupuesto, se han tomado en cuenta precios actuales del mercado, así como informes del [COIT]. El presupuesto ha sido dividido en dos apartados: costes de material y costes de personal; intentando así aumentar la granularidad y transparencia del presupuesto. Además se adjunta un desglose de las horas efectivas de trabajo dedicado, tanto por el personal de desarrollo, como por parte de la dirección.

5.2 Planificación

A lo largo de este capítulo se detallan las fases que componen el proyecto descrito en este documento. Además de identificar las fases en sí, se realiza un estudio minucioso de la secuenciación de las mismas así como de la relación entre ellas. Para completar el capítulo de planificación se añade un diagrama de Gantt que aporta una visión global de la distribución de las diferentes tareas.

5.2.1 Planificación del proyecto

En primer lugar, se enumeran las diferentes actividades, o fases, que componen el proyecto. Además de citarlas, se incluye información sobre la relación entre ellas así como la duración de cada una de las fases de forma independiente. A continuación se adjunta una tabla donde se muestra con más detalle la distribución del conjunto de tareas del proyecto:

Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1 Estudio del estado del arte	42 días	lun 01/10/12	mar 27/11/12	
2 Seminario Leone´s project Madrid	1 día	mié 28/11/12	mié 28/11/12	
3 Interacción con probe	10 días	jue 29/11/12	mié 12/12/12	2
4 Diseño del testbed	40 días	lun 10/12/12	vie 01/02/13	1;2
5 Estudio del estado del arte de las aplicaciones Web	3 días	lun 28/01/13	mié 30/01/13	
6 Estudio del estudio del arte de la tecnología J2EE	3 días	jue 31/01/13	dom 03/02/13	5
7 Desarrollo de la aplicación J2EE	24 días	lun 04/02/13	jue 07/03/13	6;4
8 Desarrollo: web - casos de uso	28 días	vie 08/03/13	mar 16/04/13	7;4
9 Desarrollo: infraestructura Samknows: casos de uso	54 días	mié 17/04/13	dom 30/06/13	8;4;3
10 Pruebas y análisis de resultados	106 días	lun 04/02/13	dom 30/06/13	
11 Documentación - redacción	41 días	lun 22/04/13	lun 17/06/13	

Ilustración 18. Planificación del proyecto

5.2.2 Diagrama de Gantt

El diagrama de Gantt es una herramienta, muy extendida, que tiene por objetivo mostrar, de una forma visual, el tiempo dedicado a las diferentes actividades de un proyecto a lo largo del mismo. Teniendo en cuenta el conjunto de actividades o fases descrito en el apartado anterior, se elabora el siguiente diagrama de Gantt:

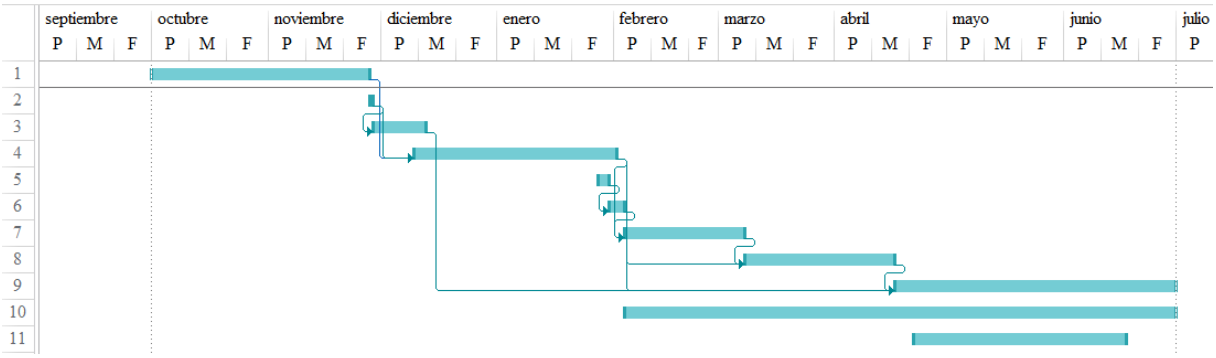

Ilustración 19. Diagrama de Gantt

5.3 Presupuesto

Una vez desglosada la planificación del proyecto, a continuación se detalla el presupuesto del mismo. Ha sido clasificado en dos categorías: costes de material y costes de personal. En la primera categoría, se incluyen materiales utilizados en los diferentes ámbitos del proyecto, como puede ser el software utilizado o el coste de las instalaciones de trabajo. En la segunda categoría se desglosan los costes derivados del personal del proyecto, diferenciando al personal encargado del desarrollo y a la dirección del mismo.

5.3.1 Costes de material

A continuación se detalla el conjunto de gastos asociados con el material destinado para el proyecto. Se ha desglosado en dos categorías: costes amortizables y costes no amortizables.

5.3.1.1 Costes amortizables

A continuación se incluye el total de costes de los materiales amortizables necesarios para la realización del proyecto descrito en este documento. Para el cálculo de la amortización, se ha seguido un modelo lineal y, se ha considerado como valor residual de dichos equipos despreciables, debido a que dichos materiales quedan obsoletos. Los datos pueden ser observados en la siguiente tabla

El coste por hora ha sido calculado tomando el valor de 1920 horas como el total de horas laborables estimadas al año. El coste final ha sido calculado tomando el valor de 720 horas como el total de horas dedicada al proyecto. Además cabe destacar, que solo se ha incluido tres licencias de software, ya que, su mayoría, se ha utilizado software de libre distribución.

	Precio (€)	Tiempo amortización (años)	Coste anual (€)	Coste / hora (€/hora)	Coste final (€)
Ordenador	1200	3	400	0,208	150
Licencia Windows 7 Professional	240	1	240	0,125	90
Licencia Windows Office 2013 Professional	539	1	539	0,28	202,13
Licencia Windows Project 2013 Professional	1369	1	1369	0,713	513,37
TOTAL					**956**

Tabla 9. Costes amortizables

5.3.1.2 Costes no amortizables

A continuación, se puede observar un desglose del conjunto de costes asociados a materiales no amortizables.

	Coste parcial / tipo	Coste total / tipo (€)
Puesto de trabajo	150 €/mes	1350
Conexión a Internet	40 €/mes	360
Costes indirectos (desplazamiento, etc.)	400 €	400
Costes varios (fotocopias, etc.)	200 €	200
Total		**2310**

Tabla 10. Costes no amortizables

5.3.1.3 Coste total de material

A continuación, se puede observar el valor total de los gastos asociados al material utilizado para el desarrollo de este proyecto. Los precios están calculados sin aplicar el IVA, el cual será añadido en el cálculo de gastos totales del proyecto más adelante:

	Costes (€)
Costes amortizables	956
Costes no amortizables	2310
Total	**3266**

Tabla 11. Coste total de material

5.3.2 Costes de personal

Para el cálculo de los costes derivados del personal se deben tener en cuenta los honorarios del equipo de desarrollo y de la dirección. La dirección del proyecto ha sido llevada a cabo por el **Dr. Francisco Valera Pintor**.

Para ajustar las cifras al trabajo realizado realmente, primero se debe calcular la cantidad total de horas dedicadas al proyecto, tanto por el personal de desarrollo, como por parte de la dirección del mismo. A continuación se muestra una tabla con los detalles del tiempo empleado a lo largo del proyecto:

	Adrián Pantoja	Francisco Valera
Duración en meses	9 meses	9 meses
Horas/semana	20 horas	3 horas
Semanas/Mes	4 semanas	4 semanas
Total	**720 horas**	**108 horas**

Tabla 12. Cómputo de horas dedicadas al proyecto

Para el cálculo de los honorarios se ha tenido en cuenta el informe del colegio de ingenieros de telecomunicaciones [COIT13] donde podemos observar, en la página 19, el sueldo medio actual para los ingenieros de telecomunicaciones. En concreto, el estudio dictamina que el sueldo medio de los ingenieros con menos de 5 años de experiencia suponen unos 25.730,70 €/año, y para los ingenieros con perfil sénior (más de 5 años de experiencia) 55.934,70 €/año.

Ajustando los salarios a dicho informe, a continuación se muestra el desglose de los salarios del personal de desarrollo y de la dirección del proyecto:

		€/año (bruto)	€/mes (bruto)	€/día (bruto)	€/hora (bruto)
Honorarios Ingeniero Técnico Teleco	Jornada completa (40h/semana)	25.730	2.144	107	**13**
Honorarios Ingeniero Superior Teleco	Jornada completa (40h/semana)	55.934	4.661	233	**29**

Tabla 13. Honorarios

Partiendo de los datos de las dos tablas anteriores, se puede elaborar una tabla con el total de los costes asociados al personal del proyecto:

Coste del personal			**Precio (€)**
Desarrollo y realización	720 horas	13 €/hora (bruto)	9.649
Dirección	108 horas	29 €/hora (bruto)	3.146
Total			**12.795**

Tabla 14. Coste total de personal

5.3.3 Presupuesto total

Teniendo en cuenta los cálculos realizados anteriormente, y el valor actual del IVA, establecido en un 21%, se obtienen los siguientes datos:

	Precio (€)
Coste del material	3.266
Coste del personal	12.795
Total (sin IVA)	16.061
Total (con IVA 21%)	**19.433**

Tabla 15. Coste final del proyecto

El presupuesto total de este proyecto asciende a la cantidad de veintidós mil ochenta y seis euros.

Leganés, a 30 de Junio de 2013

El ingeniero proyectista

Fdo. Adrián Pantoja Navarro

Capítulo 6

Conclusiones

6.1 Conclusiones

En este proyecto se ha abordado el diseño y la implementación de una herramienta que facilite la colaboración para la realización de medidas en Internet sobre el contexto del proyecto europeo Leone, dentro de su fase inicial. La consecución de este objetivo ha sido llevada a cabo mediante los siguientes subobjetivos:

- En primer lugar se ha realizado un estudio de la importancia sobre las medidas en Internet en el ámbito actual.
- A continuación se ha procedido a realizar una extensa investigación con las soluciones desarrolladas en dos ámbitos: herramientas individuales de medidas y plataformas de medidas en Internet.
- Tras finalizar la investigación, se ha intentado arrojar luz sobre los requisitos, restricciones y el marco regulador en el que se enmarca el proyecto ofreciendo datos sobre las limitaciones a las que ha sido sometido el diseño del proyecto.
- Una vez realizado todos los estudios previos, se ha procedido a realizar el diseño de la plataforma, diseñando, no solo los elementos que lo compondrían, sino también tomando en consideración los casos de uso a los que sería sometido.
- Tras definir por completo el diseño del proyecto, se ha procedido a especificar las tecnologías utilizadas para la implantación del mismo adjuntando una pequeña reseña de cada una de ellas y ofreciendo datos de configuración.

- Una vez implementado el diseño, se ha validado el sistema partiendo del componente más básico hasta la validación del sistema completo. Se han especificado tanto las pruebas realizadas, justificando el porqué de cada una de ellas, como las diferentes opciones de configuración escogidas para la realización de las mismas.
- A continuación, se ha detallado la planificación de cada uno de los procesos de las diferentes fases del proyecto, incluyendo el diagrama de Gantt, y un presupuesto completo asociado al mismo.
- Para finalizar, se ha adjuntado una completa bibliografía con artículos, documentos, y recomendaciones empleadas a lo largo de la documentación.

La herramienta desarrollada a lo largo de este proyecto, ha supuesto un reto, ya que ha sido necesario utilizar tecnologías nuevas Aunque esto puede suponer una barrera de entrada, ha resultado un incentivo para la motivación a la hora de llevar a cabo una investigación previa muy extensa.

Uno de los principales retos del proyecto, ha sido garantizar la robustez del proceso de pruebas, más allá de dejar el mismo como un simple prototipo. La validación del proyecto, y de su funcionalidad, ha sido crítica ya que se espera que sea utilizada durante los próximos dos años en el proyecto Leone.

Se ha perseguido, tanto la recolección de una buena documentación, así como el uso de técnicas y metodologías ágiles para la limpieza del código, ya que se planea seguir trabajando en este proyecto con otro personal de desarrollo en el futuro.

6.2 Posibles líneas de trabajo futuro

Las posibilidades de ampliar las funcionalidades del proyecto desplegado son innumerables. Por razones de diseño o tiempo, no han podido ser desarrolladas a lo largo de esta primera implementación. A continuación se describirán un conjunto de posibles líneas de trabajo sobre la plataforma diseñada, clasificando cada una de ellas en función del ámbito para el que puede ser provechosa:

6.2.1 Instalación de tests

Actualmente, los diferentes socios del proyecto Leone pueden aportar tests diseñados por ellos mismos al conjunto de tests ubicados en las probes lo largo de toda la infraestructura de pruebas. Los tests están compuestos por dos ficheros; un script y un binario. Al realizar la instalación de un nuevo test, éste es alojado en todas las unidades de la infraestructura de pruebas. Así surgen varias líneas de trabajo posibles:

6.2.1.1 Instalación selectiva

La instalación de tests actualmente funciona de la siguiente manera: el usuario elige qué test quiere incorporar al conjunto de tests almacenados en todos los dispositivos de la plataforma de pruebas mediante la aplicación web desarrollada y éste es almacenado en todos los dispositivos de la infraestructura.

Una opción complementaria consistiría en habilitar al usuario para que sea capaz de seleccionar en qué probes desea incorporar dicho test. Como resultado de esta acción, el resto de probes no se cargarían de tests que, a lo mejor, no son de utilidad. Además, se ganaría control en el proceso de instalación de tests.

6.2.1.2 Permisos

Se trata de una ampliación de funcionalidad a la instalación de tests selectiva. Consiste en implementar un mecanismo de permisos, o invitaciones, para limitar la instalación de tests en probes entre los diferentes usuarios. Una posibilidad a la hora de realizar la implementación sería utilizar notificaciones para comunicar a los usuarios entre sí. De esta manera, cuando uno de ellos desee instalar un nuevo test en alguna de las probes de otro usuario, deberá solicitarle la autorización para poder hacer efectiva la instalación.

6.2.1.3 Visibilidad

Una consecuencia directa de la instalación de tests selectivos es que el conjunto de tests alojados en cada una de las probes no será el mismo. Con el desarrollo actual, todos los usuarios conocen qué tests tienen alojados el resto de usuarios. Puede llegar a ser de utilidad añadir visibilidad a los tests, consiguiendo así que algunos tests sean visibles para todos y otros solo para aquellos que los instalen en sus probes, y no deseen compartir esa información con el resto. Como solución, se propone implementar algún tipo de mecanismo que aporte esta característica.

6.2.1.4 Tests complejos

El desarrollo actual solo permite tests compuestos por dos ficheros: un archivo script de control y un archivo binario. Se entiende que en el desarrollo de tests puede llegar a ser de utilidad implementar tests más complejos que se apoyen en el uso de varios ficheros y no solo de los dos básicos. Se recomienda diseñar un mecanismo que complete el proceso de instalación de tests para habilitar que el usuario pueda instalar dichos tipos de test.

6.2.1.5 Versiones

Se prevé que, a lo largo de los próximos años, los tests sean utilizados de forma continuada. Por lo tanto, éstos pueden evolucionar, y los desarrolladores pueden desear mejorarlos. Para minimizar el espacio ocupado por los tests en las diferentes unidades de la arquitectura de medidas y aumentar la optimización del sistema se recomienda utilizar algún mecanismo de gestión de versiones para los tests.

6.2.1.6 Seguridad

Se recomienda la implementación de algún mecanismo que asegure de forma eficiente la instalación de nuevos tests en las unidades de la infraestructura de medidas. Una utilidad es la erradicación, por parte de dichos mecanismos, de tests duplicados. Una primera idea es aplicar códigos hash a los ficheros que componen los tests para evitar así posibles plagios.

6.2.2 Ejecución de tests

Para un usuario puede resultar útil ejecutar un test concreto en alguna de las probes de forma instantánea utilizando la aplicación web desarrollada. Por ello, se considera que puede ser de gran utilidad si se implementa esta funcionalidad añadida.

6.2.3 Eliminación de tests

En referencia a la eliminación de tests obsoletos, se ha implementado un sistema sencillo donde el test es eliminado de todos los dispositivos una vez el usuario lo indica en la plataforma web. Surge la posibilidad de que un usuario desee eliminar un test de algún dispositivo concreto, en vez de eliminarlo de todos.

Además, surge un problema derivado de la planificación de tests: si no se modifica la planificación de un test eliminado el sistema seguirá intentando ejecutar el test dando por resultado errores. Se recomienda implementar un mecanismo que automatice la des planificación de los tests.

6.2.4 Planificación

En cuanto a la posible planificación de tests surgen varias posibles líneas de trabajo, donde destacan: el diseño de un sistema de versiones de los planificadores para los usuarios, la implementación del caso de uso denominado “Cross Probing” (el cual fue descrito en el apartado de diseño) o facilitar la planificación de dos tests para que sean ejecutados al mismo tiempo en la misma unidad.

También surge la necesidad de diseñar un sistema de des-planificación de test, así como introducir el elemento de la periodicidad en el contexto de la planificación para facilitar su uso por parte de los usuarios.

6.2.5 Resultados

Actualmente, los resultados son enviados a un repositorio central, donde son almacenados. La empresa desarrolladora de la infraestructura de pruebas, Samknows, ofrece una interfaz web para ver los resultados y estadísticos correspondientes. Se recomienda realizar un sistema similar para la aplicación web descrita en este documento.

Además, surge la posibilidad de que los usuarios deseen enviar los resultados a otro repositorio. Para lo cual se debería diseñar e implementar algún mecanismo que automatice el proceso por el cual un usuario pudiera planificar la ejecución de tests y decidir dónde van a ser enviados los resultados.

6.2.6 Gestión de usuarios

En principio, el proyecto Leone no tiene previsto crecer en términos de usuarios, aunque se recomienda implementar un mecanismo para ofrecer la posibilidad de crear nuevos usuarios en el sistema por si en un futuro se considerase relevante.

Glosario

AJAX	*Asynchronous JavaScript And XML*
API	*Application Programming Interface*
ARPANET	*Advanced Research Projects Agency Network*
AS	*Autonomous System*
ATMEN	*A triggered network measurement infrastructure*
BT	*British Telecom*
CAIDA	*Cooperative Association for Internet Data Analysis*
CDN	*Content Delivery Network*
CPU	*Central Processing Unit*
CSS	*Cascading Style Sheets*
COIT	*Colegio Oficial de Ingenieros de Telecomunicación*
DCS	*Distributed Control System*
DNS	*Domain Name System*
DTD	*Document Type Definition*
FPS	*Frames Per Second*
FTP	*File Transfer Protocol*
FTTP	*Fiber To The Premises*
HTTP	*Hypertext Transfer Protocol*
GNU	*GNU's Not Unix!*

GPL	*General Public License*
HEP	*High Energy Physics*
HTML	*HyperText Markup Language*
HTTP	*Hypertext Transfer Protocol*
I/O	*Input / Output*
ICFA	*International Committee for Future Accelerators*
IDE	*Integrated Development Environment*
IEPM-BW	*Internet End-to-end Performance Monitoring - Bandwidth to the World*
IETF	*Internet Engineering Task Force*
IMC	*Internet Measurement Conference*
IMEI	*International Mobile Equipment Identity*
IP	*Internet Protocol*
ISP	*Internet Service Provider*
IVA	*Impuesto sobre el Valor Añadido*
J2EE	*Java Platform Enterprise Edition*
JDBC	*Java Database Connectivity*
JDK	*Java Development Kit*
JSON	*JavaScript Object Notation*
JSP	*JavaServer Page*
M-Lab	*Measurement Lab*
MAC	*Media Access Control*
MAWI	*Measurement and Analysis on the WIDE Internet*
MIB	*Management Information Base*
MIT	*Massachusetts Institute of Technology*
MRTD	*Multi-threaded Routing Daemon*
NAT	*Network Address Translation*
NLANR	*National Laboratory for Advanced Network Research*
LMAP	*Large-Scale Measurement of Broadband Performance*
LSA	*Link-State Advertisement*

OS	*Operating System*
OSPF	*Open Shortest Path First*
P2P	*Peer-to-peer*
PEAR	*PHP Extension and Application Repository*
PHP	*PHP: Hypertext Preprocessor*
RADclock	*Robust Absolute and Difference Clock*
RFC	*Request For Comments*
RIPE	*Réseaux IP Européens*
RIPE NCC	*Réseaux IP Européens Network Coordination Centre*
RIS	*Research Information Systems*
RMON	*Remote Network MONitoring*
RTFM	*Real Time Flow Meetering*
RTT	*Round-trip Time*
SCIC	*Standing Committee on Inter-Regional Connectivity*
SCP	*Secure Copy*
SIGMETRICS	*Special Interest Group Metrics*
SNMP	*Simple Network Management Protocol*
SQL	*Structured Query Language*
SSL	*Secure Sockets Layer*
STREAMS	*The Stanford Data Stream Management System*
TCP	*Transmission Control Protocol*
TTM	*Test Traffic Measurement Service*
UC3M	*Universidad Carlos III de Madrid*
UCSD	*University of California, San Diego*
UDP	*User Datagram Protocol*
UK	*United Kingdom*
URL	*Uniform Resource Locator*
VoIP	*Voice Over IP*
WEB-DAV	*Web Distributed Authoring and Versioning*
XML	*Extensible Markup Language*

Referencias

[ABB+05] A. Arasu, B. Babcock, S. Babu, J. Cieslewicz, M. Datar, K. Ito, R. Motwani, U. Srivastava, and J. Widom. *STREAM: The Stanford data stream management system*. En M.Garofalakis, J. Gehrke, and R. Rastogi, editores, *Data-Stream Management – Processing High-Speed Data Streams*. Springer-Verlag, New-York, 2005

[ABD+12] *Open Source Plattforms for Inernet Monitoring and Measurement*. Giuseppe Aceto, Alessio Botta, Walter de Donato, Pietro Marchetta, Antonio Pescapé, Giorgio Ventre, University of Napoli "Federico II". Octubre 2012. Disponible [Internet]: <http://wpage.unina.it/walter.dedonato/pubs/onm_sitis12.pdf> [21 de junio de 2013]

[AIM+02] K. Anagnostakis, S.Ioannidis, S. Miltchev, J. Ioannidis, M. Greenwald, and J. Smith. *Efficient packet monitoring for network management*. 8th IEEE/IFIP Network Operations and Management Symposium (NOMS), April 2002. Disponible [Internet]: <http://repository.upenn.edu/cgi/viewcontent.cgi?article=1041&context=cis_papers> [21 de junio de 2013]

[AJAX] *W3Schools: AJAX Introduction.* Disponible [Internet]: <http://www.w3schools.com/ajax/ajax_intro.asp> [21 de junio de 2013]

[Aka] *Akamai*. Disponible [Internet]: <http://www.akamai.com> [21 de junio de 2013]

[AkcTW97] J. Apisdorf, k claffy, K. Thompson, and R. Wilder. *OC3MON: Flexible, affordable, high-performance statistics collection*. INET '97, June 1997.

[Ark] *Archipelago Measurement Infrastructure.* Disponible [Internet]: <http://www.caida.org/projects/ark/> [21 de junio de 2013]

[AS] Roy Arends and Jakob Schlyter. *Finger printing DNS servers*. Disponible [Internet]: <http://www.rfc.se/fpdns/> [21 de junio de 2013]

[Atlauc] *RIPE Atlas – Analyses and Use Cases.* Disponible [Internet]: <https://atlas.ripe.net/results/analyses/> [21 de junio de 2013]

[Atlcut] *RIPE Atlas – Comparing TCP and UDP Response Times of DNS Root Servers.* Disponible [Internet]: < https://labs.ripe.net/Members/bwijnen/tcp-udp-dns-soa-rt-ratio> [21 de junio de 2013]

[Atlg] *RIPE Atlas - results graphs.* Disponible [Internet]: <https://atlas.ripe.net/results/graphs/> [21 de junio de 2013]

[Atlgi] *RIPE Atlas – Get Involved.* Disponible [Internet] <https://atlas.ripe.net/get-involved/> [21 de junio de 2013]

[Atlif] *RIPE Atlas – A Case Study of IPv6 /48 Filtering.* Disponible [Internet]: <https://labs.ripe.net/Members/emileaben/ripe-atlas-a-case-study-of-ipv6-48-filtering> [21 de junio de 2013]

[Atlma] *RIPE Atlas – Map Visualisations.* Disponible [Internet]: <https://atlas.ripe.net/results/maps/> [21 de junio de 2013]

[Atlme] *RIPE Atlas – RIPE NCC Members.* Disponible [Internet]: <https://atlas.ripe.net/get-involved/members/> [21 de junio de 2013]

[Atludm] *RIPE Atlas – User Defined Measurements.* Disponible [Internet]: <https://atlas.ripe.net/doc/udm> [21 de junio de 2013]

[Bad] *Internet Measurements*. Badri Nath. Universidad de Rutgers, Massachusetts. Disponible [Internet]: <http://www.cs.rutgers.edu/~badri/552dir/notes/w8meas-one.pdf> [21 de junio de 2013]

[Bar04] *Measurement, Modeling and Analysis of the Internet*. Paul Barford. Universidad Wisconsin Madison. Marzo 2004. Disponible [Internet]: <http://pages.cs.wisc.edu/~pb/ima_tutorial.pdf> [21 de junio de 2013]

[Bit] *The official BitTorrent home page*. Disponible [Internet]: <http://bittorrent.com/> [21 de junio de 2013]

[BOUC] *The Legion of the Bouncy Castle.* Disponible [Internet]: <http://www.bouncycastle.org/> [21 de junio de 2013]

[Bra05a] S. Bradner, Ed. *IETF Rights in Contributions*. IETF, RFC 3978. Marzo 2005. Disponible [Internet]: <http://tools.ietf.org/pdf/bcp78.pdf> [21 de junio de 2013]

[Bra05b] S. Bradner, Ed. Intellectual Property Rights in IETF Technology. IETF. Marzo 2005. Disponible [Internet]: <http://tools.ietf.org/pdf/bcp79.pdf> [21 de junio de 2013]

[CAIDA] Cooperative Association for Internet Data Analysis. Disponible [Internet]: < http://www.caida.org/> [21 de junio de 2013]

[Cis] *Netflow services solutions guide*. Disponible [Internet]: <http://www.cisco.com/en/US/docs/ios/solutions_docs/netflow/nfwhite.html > [21 de junio de 2013]

[CJs03] Chuck Cranor, Theodore Johnson, and Oliver Spatscheck. *Gigascope: how to monitor network traffic 5Gbit/sec at a time*. Workshop on Monitoring and Processing of Data Streams (MPDS). Junio 2003.

[CJSS03a] C. Cranor, I. Johnson, O. Spatscheck, and V. Shkapenyuk. *The Gigascope stream database*. IEEE Data Engineering Bulletin, 26(1):27-32. 2003. Disponible [Internet]: <http://sites.computer.org/debull/A03mar/A03MAR-CD.pdf#page=29> [21 de junio de 2013]

[CJSS03b] C.D. Cranor, T.Johnson, O. Spatscheck, and V. Shkapenyuk. *Gigascope stream database for network applications*. ACM SIGMOD, páginas 647-651. Junio 2003. Disponible [Internet]: < http://dl.acm.org/citation.cfm?id=872838> [21 de junio de 2013]

[CK06] M. Crovella, B. Krishnamurthy. *Internet Measurement: Infrastructure, Traffic and Applications.* Wiley. ISBN: 0-470-01461-X. 2006

[CMC05] M. Crovella, J. Micheel and K. Claffy. *Workshop on Community Oriented Network Measurement Infrastructure*. Passive and Active Measurement (PAM 2005) Workshop, Boston, Massachusetts. Marzo 2005. Disponible [Internet]: <http://www.caida.org/workshops/conmi/0503/> [21 de junio de 2013]

[CMM+04] J.Cuellas, J. Morris, D. Mulligan, J. Peterson, and J. Polk. *Geopriv requirements*. RFC 3693. Febrero 2004. Disponible [Internet]: < http://www.ietf.org/rfc/rfc3693.txt> [21 de junio de 2013]

[COIT] Colegio Oficial de Ingenieros de Telecomunicación. Disponible [Internet]: <http://www.coit.es> [21 de junio de 2013]

[COIT13] *El Ingeniero de Telecomunicación: Perfil Socio-Profesional*. Colegio Oficial de Ingenieros de Telecomunicación. Febrero 2013. Disponible [Internet]: <http://www.coit.es/descargar.php?idfichero=5985> [21 de junio de 2013]

[COA] *Co-advisor test suite*. Disponible [Internet]: <http://coad.measurement-factory.com/> [21 de junio de 2013]

[Cor] *Coralreef – supported hardware*. Disponible [Internet]: <http://www.caida.org/tools/measurement/coralreef/hardware.xml> [21 de junio de 2013]

[Cot05a] Les Cottrell. *ICFA SCIC network monitoring report*. Stanford Linear Accelerator Center. Febrero 2005. Disponible [Internet]: <http://www.slac.stanford.edu/xorg/icfa/icfa-net-paper-jan05/> [21 de junio de 2013]

[Cot05b] Les Cottrell. *PingER*. 2005. Disponible [Internet]:<http://www-iepm.slac.stanford.edu/pinger/> [21 de junio de 2013]

[CRWDD] *Community Resource for Archiving Wireless Data At Dartmouth*. Disponible [Internet]: <http://crawdad.cs.dartmouth.edu> [21 de junio de 2013]

[DNSc] *DNSchecker – squishywishywoo: Complete DNS traversal checking*. Disponible [Internet]: <http://www.squish.net/dnscheck> [21 de junio de 2013]

[Dnsd] *DSC: A DNS statistics collector*. Disponible [Internet]: <http://dns.measurement-factory.com/tools/dsc/> [21 de junio de 2013]

[Dnsf] *Dsnsstat*. Disponible [Internet]: <http://www.caida.org/tools/utilities/dnsstat/> [21 de junio de 2013]

[Dnsg] *DNSTOP: stay on top of your DNS traffic*. Disponible [Internet]: <http://www.caida.org/tools/utilities/dnstop/> [21 de junio de 2013]

[Duf04] Dick Duffield. *Sampling for Passive Internet Measurement: a Review*. Instituto matemáticas estadísticas, Vol.19, No. 3, 472-498. 2004. Disponible [Internet]: <http://www2.research.att.com/~duffield/papers/STS102.pdf> [21 de junio de 2013]

[EBM13] P. Eardley, T. Burbridge, A. Morton. *A framework for large-scale measurements*. IETF. Febrero 2013. Disponible [Internet]: < http://tools.ietf.org/html/draft-eardley-lmap-framework-01 > [21 de junio de 2013]

[ECL] *Eclipse.* Disponible [Internet]: <http://www.eclipse.org/> [21 de junio de 2013]

[ECLb] *Eclipse – FAQ: What is Eclipse?* Disponible [Internet]: <http://wiki.eclipse.org/FAQ_What_is_Eclipse%3F> [21 de junio de 2013]

[ERM] *ERMaster*. Disponible [Internet]: <http://ermaster.sourceforge.net/> [03 de junio de 2013]

[EMB+13] P. Eardley, A. Morton, M. Bagnulo, T. Burbridge. *Terminology for Large MeAsurement Platforms (LMAP).* IETF. Mayo 2013. Disponible [Internet]: <http://tools.ietf.org/html/draft-eardley-lmap-terminology-01> [21 de junio de 2013]

[Eva] *Evalid WebSite analysis and testing suite*. Disponible [Internet]: <http://www.soft.com/eValid/> [21 de junio de 2013]

[Fer10] *Pruebas de integración orientadas a objetos.* Juan Manuel Fernández Peña. 2010. Disponible [Internet]: <http://www.uv.mx/personal/jfernandez/files/2010/07/Pruebas-de-Integracion.pdf> [21 de junio de 2013]

[FK05] Glenn Fowler and Balachander Krishnamurthy. *Dss-datastream and scan*. AT&T Labs-Research. Agosto 2005. Disponible [Internet]: < http://www2.research.att.com/~gsf/publications/dss-2004.pdf> [21 de junio de 2013]

[Ful] Mark Fullner. *Flowtools*. Disponible [Internet]: <http://www.splintered.net/sw/flow-tools/> [21 de junio de 2013]

[FULLC] *FullCalendar*. Disponible [Internet]: <http://arshaw.com/fullcalendar/> [21 de junio de 2013]

[FV02] Mike Fisk and George Varghese. *Agile and scalable analysis of network events*. ACM SIGCOMM Internet Measurement Workshop, páginas 285-290, Marsella. Noviembre 2002. Disponible [Internet]: <http://dl.acm.org/citation.cfm?id=637245> [21 de junio de 2013]

[Gam] *GameSpy: Gamig's home page*. Disponible [Internet]: <http://www.gamespy.com> [21 de junio de 2013]

[gnu] *GNU zebra*. Disponible [Internet]: <http://www.zebra.org> [21 de junio de 2013]

[Gom] Gomez Internet performance management. Disponible [Internet]: < http://www.compuware.com/application-performance-management/gomez-apm-products.html> [21 de junio de 2013]

[GP03] Jose Maria Gonzalez and Vern Paxson. *Pktd: Apacket capture and injection daemon*. Passive and Active Measurement Workshop, La Jolla, CA. Abril 2003. Disponible [Internet]: < http://www.bandwidthco.com/whitepapers/netforensics/crafting/pktd%20-%20A%20Packet%20Capture%20and%20Injection%20Daemon.pdf> [21 de junio de 2013]

[Gro95] *Roads and Crossroads of the Internet History*. Gregory Gromov. 1995. Disponible [Internet]: < http://www.netvalley.com/history_of_internet.html> [21 de junio de 2013]

[HBB02] Khaled Harfoush, Azer Bestavros, and John Byers. *PeriScope: An active measurement API*. Pasive and Active Measurement Workshop. Marzo 2002. Disponible [Internet]: < http://www.cs.bu.edu/fac/best/res/papers/pam02.pdf> [21 de junio de 2013]

[HEP] *High Energy Physics.* Disponible [Internet]: <http://science.energy.gov/hep/> [21 de junio de 2013]

[Hostc] *Hostcount*. Disponible [Internet]: <http://www.hostcount.com/> [21 de junio de 2013]

[HPMkc02] B. Huffaker, D. Plummer, D. Moore, and k claffy. *Topology discovery by active probing*. Applications and the Internet (SAINT). 2002. Disponible [Internet]: < http://www.caida.org/publications/papers/2002/SkitterOverview/> [21 de junio de 2013]

[ICANN] *Internet Corporation for Assigned Names and Numbers*. Disponible [Internet]: <http://www.icann.org/> [21 de junio de 2013]

[ICFA] *International Committee for Future Accelerators*. Disponible [Internet]: <http://www.fnal.gov/directorate/icfa/> [21 de junio de 2013]

[IETF] *The Internet Engineering Task Force (IETF)*. Disponible [Internet]: <http://www.ietf.org/> [21 de junio de 2013]

[IETFaid] IETF - Active Internet-Drafts. Disponible [Internet]: <http://datatracker.ietf.org/doc/active/> [21 de junio de 2013]

[IEPM-BW] *Internet End-to-end Performance Monitoring - Bandwidth to the World (IEPM-BW) project.* Disponible [Internet]: <http://www-iepm.slac.stanford.edu/bw/> [21 de junio de 2013]

[IMC] *Internet Measurement Conference*. Disponible [Internet]: <http://www.sigcomm.org/events/imc-conference/> [21 de junio de 2013]

[Integ] *Pruebas de Integración.* Disponible [Internet]: <http://200.69.103.48/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node31.html> [21 de junio de 2013]

[Iperf] Disponible [Internet]: <http://iperf.fr/> [21 de junio de 2013]

[IWS] *Internet World Stats*. Miniwatts Marketing Group. Disponible [Internet]: < http://www.internetworldstats.com/> [21 de junio de 2013]

[J2EE] *The Java EE 5 Tutorial.* Oracle. Disponible [Internet]: <http://docs.oracle.com/javaee/5/tutorial/doc/gfirp.html> [21 de junio de 2013]

[JAVA] *¿Qué es Java?* Oracle. Disponible [Internet]: <http://www.java.com/es/download/whatis_java.jsp> [21 de junio de 2013]

[JAVAb] *New to Programming Center*. Oracle. Disponible [Internet]: <http://www.oracle.com/technetwork/topics/newtojava/overview/index.html> [21 de junio de 2013]

[JAVAc] *Descarga Gratuita de Java*. Oracle. Disponible [Internet]: <http://www.java.com/es/download/> [21 de junio de 2013]

[JAVAd] *Java Development Kit 7 Downloads*. Oracle. Disponible [Internet]: <http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html?ssSourceSiteId=otnes> [21 de junio de 2013]

[JMSS05] Theodore Johnson, S. Muthukrishnam, Oliver Spatscheck, and Divesh Srivastava. *Streams, security and scalability*. 19th IFIP WG11.3 Working Conference on Data and Application Secutiry, vol. 1654 de LNCS, páginas 1-15. Agosto 2005. Disponible [Internet]: <http://dl.acm.org/citation.cfm?id=2138933> [21 de junio de 2013]

[JQ] *jQuery: Write less, do more.* Disponible [Internet]: <http://jquery.com/> [21 de junio de 2013]

[JSON] *JSON: JavaScript Object Notation.* Disponible [Internet]: <http://www.json.org/> [21 de junio de 2013]

[JZL] *JZlib: zlib in pure Java.* Disponible [Internet]: <http://www.jcraft.com/jzlib/> [21 de junio de 2013]

[KBV+04] M. Kaart, J.P. van Best, W. Vree y L. Torenvliet. *The importance of measurements for Internet policy*. IEEE 2004 International Conference on Systems, Man, and Cybernetics (SMC2004) 4711-4716. The Hague, The Netherlands. Octubre 2004. Disponible [Internet]: <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1401275> [21 de junio de 2013]

[KCC+03] Sailesh Krishnamurthy, Sirish Chandrasekaran, Owen Cooper, Amol Deshpande, Michael J. Franklin, Joseph M. Hellerstein, Wei Hong, Samuel R. Madden, Vijayshankar Raman, Fred Reiss, and Mehul A. Shah. *TelegraphCQ: An architectural status report*. IEEE Data Engineering Bulletin, 26(1):11-18, 2003. Disponible [Internet]: < http://www.mathcs.emory.edu/~cheung/papers/StreamDB/Systems/tcqdebulletin.pdf> [21 de junio de 2013]

[Keya] *Keynote*. Disponible [Internet]: <http://keynote.com> [21 de junio de 2013]

[Keyb] Ken Keys. *Iffinder software* Disponible [Internet]: <http://www.caida.org/tools/measurement/iffinder/> [21 de junio de 2013]

[KMK+01] Ken Keys, David Moore, Ryan Koga, Edouard Lagache, Michael Tesch, and k claffy. *The architecture of CoralReef; an Internet traffic monitoring software suite.* Passive and Active Measurement Workshop. CAIDA. Abril 2001. Disponible [Internet]: <http://www.caida.org/tools/measurement/coralreef/> [21 de junio de 2013]

[KMS05] Balachander Krishnamurthy, Harsha Madhyastha, and Oliver Spatscheck, *Atmen: A triggered network measurement infrastructure*. World Wide Web Conference 2005.

Mayo 2005. Disponible [Internet]: < http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.68.1031> [21 de junio de 2013]

[KMV05] Balachander Krishnamurthy, Harsha Madhyastha, and Suresh Venkatasubramanian. *On stationarity in Internet measurements through an information-theoretic lens*. Network Database Workshop. Abril 2005. Disponible [Internet]: < http://www2.research.att.com/~bala/papers/atmen-netdb.pdf> [21 de junio de 2013]

[Koh] Eddie Kohler. *Ipsumdump*. Disponible [Internet]: <http://www.cs.ucla.edu/~kohler/ipsumdump/> [21 de junio de 2013]

[Koh06] Eddie Kohler. *Click for measurement*. TR060010, Deparment of Computer Science, UCLA. Febrero 2006. Disponible [Internet]: <http://read.seas.harvard.edu/~kohler/pubs/kohler06click.pdf> [21 de junio de 2013]

[KVV06] *The importance of Internet topology measurements*. Marnix Kaart, Jos Vranken y Wim Vree. Publicación de Delft University of Technology, Jaffalaan 5, Delft, The Netherlands. Julio 2006. Disponible [Internet]: <http://userpage.fuberlin.de/~jmueller/its/conf/amsterdam06/downloads/papers/Kaart_Vranckeen_Vree.pdf> [21 de junio de 2013]

[KW00] Balachander Krishnamurthy and Jia Wang. *On network-aware clustering of Web clients*. ACM Sigcomm. Agosto 2000. Disponible [Internet]: < http://dl.acm.org/citation.cfm?id=347412> [21 de junio de 2013]

[LEONE] *Leone: From global measurements to local management*. Disponible [Internet]: <http://cordis.europa.eu/search/index.cfm?fuseaction=proj.document&PJ_RCN=13391877> [03 de junio de 2013]

[MAWI] *Measurement and Analysis on the WIDE Internet*. Disponible [Internet]: < http://www.wide.ad.jp/project/wg/mawi.html > [21 de junio de 2013]

[MCC04] Matthew L. Massie, Brent N. Chun, and David E. Culler. *The Ganglia distributed monitoring system: Design, implementation, and experience*. Parallel Computing, 21(7). Julio 2004. Disponible [Internet]: <http://www.ittc.ku.edu/~niehaus/classes/750-s07/documents/ganglia-parallel-computing.pdf> [21 de junio de 2013]

[MHK+03] Andrew Moore, James Hall, Christian Kreibich, Euan Harris, and Ian Pratt. *Architecture of a network monitor*. Passive and Active Measurement Workshop, La Jolla, CA. Abril 2003. Disponible [Internet]: <http://vodun.org/papers/net-papers/moore_architecture_of_a_network_monitor.pdf> [21 de junio de 2013]

[MJ98a] G. Robert Malan and Farnam Jahanian. *An extensible probe architecture for network protocol performance measurement*. ACM SIGCOMM. 1998. Disponible [Internet]: <http://dl.acm.org/citation.cfm?id=285243.285284> [21 de junio de 2013]

[MLAB] *Measurement Lab*. Disponible [Internet]: <http://measurementlab.net> [21 de junio de 2013]

[Mog90] J. Mogul. *Efficient use of workstations for passive monitoring of local area networks*. ACM SIGCOMM, páginas 253-263, ACM Press. 1990. Disponible [Internet]: <http://dl.acm.org/citation.cfm?id=99562> [21 de junio de 2013]

[MR] K. McCloghrie and M.Rose. *Management information base for betwork management of TCP/IP-based internets: MIB-II*. IETF STD 17, RFC 1213. Disponible [Internet]: <http://www.ietf.org/rfc/rfc1213.txt> [21 de junio de 2013]

[MRTa] *Multi-threaded routing toolkit*. Disponible [Internet]: <http://tools.ietf.org/html/rfc6396> [21 de junio de 2013]

[MyS] *MySQL: Why MySQL?* Oracle. Disponible [Internet]: <http://www.mysql.com/why-mysql/> [03 de junio de 2013]

[Nap] *Napster*. Disponible [Internet]: <http://www.napster.com> [21 de junio de 2013]

[Nete] *Netrament – a network traffic flow measurement tool*. Disponible [*Internet*]: <http://www.caida.org/tools/measurement/netramet/> [21 de junio de 2013]

[NLANR] *National Laboratory for Applied Network Research*. 2006. Disponible [Internet]: < http://www.nlanr.net/> [21 de junio de 2013]

[PACR02] Larry Peterson, Tom Anderson, David Culler, and Timothy Roscoe. *A blueprint for introducing disruptive technology into the Internet*. HotNets-I. Octubre 2002. Disponible [Internet]: <http://www.cs.princeton.edu/courses/archive/fall03/cs597B/handouts/pdn02-001.pdf> [21 de junio de 2013]

[PAMM98] Ven Paxson, Guy Almes, Jamsheed Mahdavi, and Matt Mathis. *Framework for IP performance metrics*. RFC 2330. Mayo 1998. Disponible [Internet]: < http://tools.ietf.org/html/rfc2330> [21 de junio de 2013]

[Peu02] *Internet traffic measurements aims, methodology, and discoveries*. Markus Peuhkuri. Tesis en Universidad de Tecnología de Helsinki. Mayo 2002. Disponible [Internet]: <https://www.netlab.tkk.fi/~puhuri/publications/li.pdf> [21 de junio de 2013]

[PGB+04] Thomas Plagemann, Vera Goebel, Andrea Bergamini, Giacomo Tolu, Guillaume UrvoyKeller, y Ernst W. Biersack. *Using data stream management systems for traffic analysis – a case study*. Passive and Active Measurement Workshop, páginas 215-226. 2004. Disponible [Internet]: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.118.7737> [21 de junio de 2013]

[PHPMA] *phpMyAdmin: Bringing MySQL to the web*. Disponible [Internet]: <http://www.phpmyadmin.net/home_page/index.php> [03 de junio de 2013]

[PHPMAb] *phpMyAdmin: Documentation.* Disponible [Internet]: <http://www.phpmyadmin.net/home_page/docs.php> [03 de junio de 2013]

[PHPMAc] *phpMyAdmin: Welcome to phpMyAdmin Wiki.* Disponible [Internet]: <http://wiki.phpmyadmin.net/pma/Welcome_to_phpMyAdmin_Wiki> [03 de junio de 2013]

[PN99] Ram Periakaruppan and Evi Nemeth. *GTrace – a graphical traceroute*. 13th USENIX conference on System administration, páginas 69-78, Berkeley, CA, USA. 1999. Disponible [Internet]: <http://www.caida.org/tools/visualization/gtrace/> [21 de junio de 2013]

[Qst] *Real-time game server status*. Disponible [Internet]: <http://www.clusterresources.com/torquedocs21/commands/qstat.shtml> [21 de junio de 2013]

[quaa] *The Quagga routing suite*. Disponible [Internet]: <http://www.quagga.net> [21 de junio de 2013]

[R10] *The Language of SQL*. Larry Rockoff. Course Technology PTR. 2010. Disponible [Internet]: <http://proquest.safaribooksonline.com/book/databases/sql/9781435457515> [03 de junio de 2013]

[RADc] *RADclock project.* Disponible [Internet]: <http://www.synclab.org/radclock/> [21 de junio de 2013]

[RAE01] *Diccionario de la Lengua Española*. Real Academia de la Lengua Española. Edición 22ª. 2001. ISBN: 9788423968145. Disponible [Internet]: <http://www.rae.es> [21 de junio de 2013]

[RFC4353] *RFC 4353 A framework for Conferencing with the Session Initiation Protocol (SIP).* J. Rosenberg. Febrero 2006. Disponible [Internet]: <http://www.ietf.org/rfc/rfc4353.txt> [21 de junio de 2013]

[RIPEm] *RIPE Mailing Lists.* Disponible [Internet]: <http://www.ripe.net/ripe/mail> [21 de junio de 2013]

[RIPEme] *RIPE Meetings.* Disponible [Internet]: <http://www.ripe.net/ripe/meetings/faq-meetings/ripe-meeting-faqs> [21 de junio de 2013]

[RIPE-NCC] *RIPE Network Coordination Centre.* Disponible [Internet]: <http://www.ripe.net/> [21 de junio de 2013]

[RIPEwg] *RIPE Work Groups.* Disponible [Internet]: <http://www.ripe.net/ripe/groups/wg> [21 de junio de 2013]

[RIS] *Routing Information Service (RIS).* RIPE NCC. Disponible [Internet]: <http://www.ripe.net/data-tools/stats/ris/routing-information-service> [21 de junio de 2013]

[Rou] *Routeviews Project*. Disponible [Internet]: <http://www.routeviews.org> [21 de junio de 2013]

[RS] Route Servers. Disponible [Internet]: <http://root-servers.org/> [21 de junio de 2013]

[Sca] Scamper. Disponible [Internet]: <http://www.wand.net.nz/scamper> [21 de junio de 2013]

[SCIC] Standing Committee on Inter-Regional Connectivity (SCIC). Disponible [Internet]: <http://icfa-scic.web.cern.ch/icfa-scic/> [21 de junio de 2013]

[SCRa] *Scrum Manager: Pruebas Unitarias.* Disponible [Internet]: <http://www.scrummanager.net/bok/index.php?title=Pruebas_unitarias> [21 de junio de 2013]

[SG01] Aman Shaikh and Albert Greenberg. *Experience in black-box OSPF measurement*. ACM SIGCOMM Internet Measurement Workshop. Noviembre 2001. Disponible [Internet]: <http://dl.acm.org/citation.cfm?id=505218> [21 de junio de 2013]

[SIG] *ACM SIGMETRICS*: *Special interest group on performance evaluation*. Disponible [Internet]: <http://www.sigmetrics.org/> [03 de junio de 2013]

[SIG+02] Aman Shaikh, Chris Isett, Albert Greenberg, Matthew Roughan, and Joel Gottlieb. *A case study of OSPF behavior in a large enterprise network*. ACM SIGCOMM Internet Measurement Workshop, páginas 217-230, Marsella. Noviembre 2002. Disponible [Internet]: <http://dl.acm.org/citation.cfm?id=637236> [21 de junio de 2013]

[SJM12] H. Schulzrinne, W. Johnston, J. Miller. *Large-Scale Measurement of Broadband Performance: Use Cases,Architecture and Protocol Requirements*. IETF. Septiembre 2012. Disponible [Internet]: <http://tools.ietf.org/html/draft-schulzrinne-lmap-requirements-00> [21 de junio de 2013]

[SLF4J] *Simple Logging Facade for Java.* Disponible [Internet]: <http://www.slf4j.org/> [21 de junio de 2013]

[SM04] *Internet Measurement Data Catalog: Motivation and Design Principles*. Colleen Shannon, David Moore. Conferencia ACM Internet Measurement (IMC 2004). Octubre 2004. Disponible [Internet]: <http://www.caida.org/publications/presentations/2004/datcat_principles_imc/datcat_principles_imc.pdf> [21 de junio de 2013]

[SMPA] *Scrum Manager: Pruebas de aceptación.* Disponible [Internet]: <http://www.scrummanager.net/bok/index.php?title=Pruebas_de_aceptaci%C3%B3n> [21 de junio de 2013]

[SMpi] *Scrum Manager: Pruebas de integración.* Disponible [Internet]: <http://www.scrummanager.net/bok/index.php?title=Pruebas_de_integraci%C3%B3n> [21 de junio de 2013]

[SMPS] *Scrum Manager: Pruebas de sistema.* Disponible [Internet]: <http://www.scrummanager.net/bok/index.php?title=Pruebas_de_sistema> [21 de junio de 2013

[SP02] Jesse Steinberg and Joseph Pasquale. *A Web middleware architecture for dynamic customization of content for wireless clients*. World Wide Web Conference. Mayo 2002. Disponible [Internet]: <http://dl.acm.org/citation.cfm?id=511529> [21 de junio de 2013]

[Spoof] *Spoofer Project: Spoofer Main.* Disponible [Internet]: <http://spoofer.csail.mit.edu/> [21 de junio de 2013]

[SPR] *Spring: What is Spring?* Spring Source. Disponible [Internet]: <http://www.springsource.org/> [21 de junio de 2013]

[SSHJ] *SSHJ*. Disponible [Internet]: <https://github.com/shikhar/sshj> [21 de junio de 2013]

[Ste] *Steam*. Disponible [Internet]: <http://www.steampowered.com> [21 de junio de 2013]

[STR] *Struts.* The Apache Software Foundation. Disponible [Internet]: <http://struts.apache.org/> [21 de junio de 2013]

[SWA03] Neil Spring, Davd Wetherall, and Tom Anderson. *Scriptroute: A public Internet measurement facility*. USENIX Symposium on Internet Technologies and Systems (USITS). 2003. Disponible [Internet]: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.132.376> [21 de junio de 2013]

[TCD03] Ajay Tirumala, Les Cottrell, and Tom Dunigan. Measuting end-to-end bandwith with Iperf using Web100. Passive and Active Measurement Workshop, La Jolla, CA. Abril 2003. Disponible [Internet]: < http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.13.1705> [21 de junio de 2013]

[TDa] *The IPv4 Routed /24 Topology Dataset.* Disponible [Internet]: <http://www.caida.org/data/active/ipv4_routed_24_topology_dataset.xml> [21 de junio de 2013]

[TDb] *The IPv6 Topology Dataset.* Disponible [Internet]: <http://www.caida.org/data/active/ipv6_allpref_topology_dataset.xml> [21 de junio de 2013]

[TLP] *Trust Legal Provisions (TLP) Documents*. IETF. Disponible [Internet]: <http://trustee.ietf.org/license-info/> [21 de junio de 2013]

[TMW97] K. Thompson, G. Miller, and R. Wilder. *Wide-area traffic patterns and characteristics*. IEEE Network, 11(6):10-23. Noviembre 1997. Disponible [Internet]: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.93.65> [21 de junio de 2013]

[TOM] *Apache Tomcat 7: Introduction*. The Apache Software Foundation. Disponible [Internet]: <http://tomcat.apache.org/tomcat-7.0-doc/introduction.html> [21 de junio de 2013]

[TOMb] *Apache Tomcat: Get Involved.* The Apache Software Foundation. Disponible [Internet]: <http://tomcat.apache.org/getinvolved.html> [21 de junio de 2013]

[TOMc] *Apache Tomcat: Tomcat 7 Downloads.* The Apache Software Foundation. Disponible [Internet]: <http://tomcat.apache.org/download-70.cgi> [21 de junio de 2013]

[TS05] Claudia Linnhoff-Popien Thomas Strang, editor. Location-and Context Awareness: First Internetional Workshop, Lecture Notes in Computer Science 3479. Springer-Vrelag GmbH. 2005

[TTM] *Test Traffic Measurement Service.* Disponible [Internet]: <https://www.ripe.net/data-tools/stats/ttm/test-traffic-measurement-service> [21 de junio de 2013]

[vdMCCS00] Jaobus van der Merwe, Ramon Caceres, Y-H Chu, and Corman Sreenan, *Mmdump – a tool for monitoring Internet multimedia traffic*. ACM Computer Communication Review, 30(4), October 2000. Disponible [Internet]: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.28.8783> [21 de junio de 2013]

[Wal00] S.Waldbusser. *Remote network monitoring management information base*. RFC 2819. 2000. Disponible [Internet]: <http://tools.ietf.org/html/rfc2819> [21 de junio de 2013]

[WCZ01] Yubin Wang, Mark Claypool, and Zheng Zuo. *An empirical study of Real Video performance across the Internet*. ACM SIGCOMM Internet Measurement Workshop. Noviembre 2001. Disponible [Internet]: <ftp://ftp.cs.wpi.edu/pub/techreports/pdf/01-16.pdf> [21 de junio de 2013]

[WMJ04] David Watson, G. Robert Malan, and Farnam Jahanian. *An extensible probe architecture for network protocol performance measurement*. *Software – Practice and Experience*, 34(1):47-67, 2004. Disponible [Internet]: <http://deepblue.lib.umich.edu/handle/2027.42/34551> [21 de junio de 2013]

[WPP+04] Limin Wang, KyoungSoo Park, Ruoming Pang, VIvek S. Pai, and Larry Peterson, *Reliability and security in the CoDeeN content distribution network*. USENIX 2004 Annual Technical Conference. Junio 2004. Disponible [Internet]: <http://dl.acm.org/citation.cfm?id=1247429> [21 de junio de 2013]

[XAMPP] *XAMPP*. Disponible [Internet]: <http://www.apachefriends.org/es/xampp.html> [21 de junio de 2013]

[ZDPS01] Zin Zhang, Nck Duffield, Vern Paxson, and Scott Shenker. *On the constancy of Internet path properties*. ACM SIGCOMM Internet Measurement Workshop. Noviembre 2001. Disponible [Internet]: <http://conferences.sigcomm.org/imc/2001/imw2001-papers/38.pdf> [21 de junio de 2013]

[ZLIB] *Zlib: A Massively Spiffy Yet Dedicately Unobtrusive Compression Library*. Disponible [Internet]: <http://zlib.net/> [21 de junio de 2013]

[ZLPG05] Han Zheng, Eng Keong Lua, Marcelo Pias, and Timothy Griffin. *Internet routing policies and round-trip-times*. Passive and Active Measurement Workshop. Abril 2005. Disponible [Internet]: < http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.98.8551> [21 de junio de 2013]

[ZZP+04] M. Zhang, C. Zhang, V. Pai, L. Peterson, and R. Wang. Planetseer: *Internet path failure monitoring and characterization in wide-area services*. Sixth Symposium on Operating Systems Designs and Implementation (OSDI ’04). 2004. Disponible [Internet]: < http://dl.acm.org/citation.cfm?id=1251266> [21 de junio de 2013]